

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	139	(717/155).ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:23
S2	27	S1 and merging	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 15:45
S3	3	S2 and (estimating and data\$1flow)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:19
S4	1	((estimating or estimate\$1)cost) and (data\$1flow analysis)) and (merged or merging) and mutex	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 09:55
S5	2	("7124271").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/03 12:51
S6	0	("compiler and (forward disjunctive dataflow analysis)").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/03 12:51
S7	4	compiler and (forward disjunctive dataflow analysis)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:59

EAST Search History

S8	6	S1 and merging and (thread\$1 or threading)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 15:46
S9	2	S1 and merging and (thread\$1 or threading) and (data\$1flow)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:00
S10	4	forward disjunctive dataflow analysis	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:14
S11	2	S1 and (critical section)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:14
S12	162	(compiler or optimization) and (estimating and data\$1flow)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:19
S13	8	(compiler or optimization) and (estimating same data\$1flow)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/03 16:28
S14	2	("7124271").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/03 16:28

EAST Search History

S15	2	("20050108695").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 10:00
S16	0	("10714198").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 10:00
S17	0	("10714198").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 10:25
S18	2	("20050108695").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 10:25
S19	139	(717/155).ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:23
S20	139	S19	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:23
S21	2	S19 and (critical (section\$1 or area\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:26

EAST Search History

S22	365	compiler and (critical (section\$1 or area\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:47
S23	5	S22 and (cost same matrix) and merge\$1	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:28
S24	1237	compiler and (critical (section\$1 or area\$1 or path\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:47
S25	164	compiler same (critical (section\$1 or area\$1 or path\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:48
S26	102	S25 and (combine\$1 or merge\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 11:48
S27	10	S25 same (combine\$1 or merge\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 14:16
S28	2	("20030208673").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 15:00

EAST Search History

S29	0	("(merge\$ormerg\$3)near4(critical section\$1)").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2007/04/13 15:01
S30	5	(merge\$1 or merg\$3) near4 (critical section\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 15:08
S31	48	(merge\$1 or merg\$3) near4 (critical (section\$1 or area\$1 or path\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 16:44
S32	242	(data\$1flow analysis) and compiler	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 16:45
S33	12	((data\$1flow analysis) same cost) and compiler	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 16:48
S34	5	((data\$1flow analysis) same cost) and compiler and matrix and vector\$1	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 17:25
S35	3	((data\$1flow analysis) same cost) and compiler and (forward disjunctive)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 17:26

EAST Search History

S36	3	((data\$1flow analysis) same cost) and (forward disjunctive)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/13 17:26
S37	4	(data\$1flow analysis) and (forward disjunctive)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 10:48
S38	4537	(assign\$3 near5 lock\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 10:49
S39	365	compiler and (critical (section\$1 or area\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 10:49
S40	365	S39	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 10:49
S41	10	S39 and (assign\$3 near3 lock\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 12:20
S42	77	S39 and (redundant)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 15:12

EAST Search History

S43	1	S39 and (redundant same enter\$1 same exit\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 12:21
S44	7	S39 and (redundant same enter\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 12:21
S45	3	S39 and (remov\$3 near3 redundant)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/04/14 15:12



Terms used

[**compiler optimization multi threading code motion critical section network processor**](#)Found **60,664** of
199,787

Sort results by

 [Save results to a Binder](#)[Try an Advanced Search](#)

Display results

 [Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale

1 Automatic parallelization: Automatic multithreading and multiprocessing of C programs for IXP

◆ Long Li, Bo Huang, Jinquan Dai, Luddy Harrison
June 2005 Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '05

Publisher: ACM PressFull text available: [pdf\(634.36 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Effective compilation of packet processing applications onto the Intel IXP network processors requires, among other things, the automatic use of multiple threads on one or more processing elements, and the automatic introduction of synchronization as required to correctly enforce dependences between such threads. We describe the program transformation that is used in the Intel Auto-partitioning C Compiler for IXP to automatically multithread/multi-process a program for the IXP. This transformati ...

Keywords: code motion, critical section, multi-processing, multi-threading, network processor

2 Effective thread management on network processors with compiler analysis

◆ Xiaotong Zhuang, Santosh Pande
June 2006 ACM SIGPLAN Notices , Proceedings of the 2006 ACM SIGPLAN/SIGBED conference on Language, compilers and tool support for embedded systems LCTES '06, Volume 41 Issue 7

Publisher: ACM PressFull text available: [pdf\(466.13 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Mapping packet processing tasks on network processor micro-engines involves complex tradeoffs that relating to maximizing parallelism and pipelining. Due to an increase in the size of the code store and complexity of the application requirements, network processors are being programmed with heterogeneous threads that may execute code belonging to different tasks on a given micro-engine. Also, most network applications are streaming applications that are typically processed in a pipelined fashion ...

Keywords: CPU scheduling, compiler optimizations, network processors, real-time scheduling

3 GPGPU: general purpose computation on graphics hardware

◆ David Luebke, Mark Harris, Jens Krüger, Tim Purcell, Naga Govindaraju, Ian Buck, Cliff Woolley, Aaron Lefohn
August 2004 ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04

Publisher: ACM PressFull text available: [pdf\(63.03 MB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#)

The graphics processor (GPU) on today's commodity video cards has evolved into an extremely powerful and flexible processor. The latest graphics architectures provide tremendous memory bandwidth and computational horsepower, with fully programmable vertex and pixel processing units that support vector operations up to full IEEE floating point precision. High level languages have emerged for graphics hardware, making this computational power accessible. Architecturally, GPUs are highly parallel s ...

4 Exploiting perception in high-fidelity virtual environments: Exploiting perception in high-fidelity virtual environments

◆ Additional presentations from the 24th course are available on the citation page

Mashhuda Glencross, Alan G. Chalmers, Ming C. Lin, Miguel A. Otaduy, Diego Gutierrez
July 2006 **ACM SIGGRAPH 2006 Courses SIGGRAPH '06**

Publisher: ACM Press

Full text available:  pdf(5.07 MB)  Additional Information: [full citation](#), [abstract](#), [references](#)
[mov\(68:6 MIN\)](#)

The objective of this course is to provide an introduction to the issues that must be considered when building high-fidelity 3D engaging shared virtual environments. The principles of human perception guide important development of algorithms and techniques in collaboration, graphical, auditory, and haptic rendering. We aim to show how human perception is exploited to achieve realism in high fidelity environments within the constraints of available finite computational resources. In this course w ...

Keywords: collaborative environments, haptics, high-fidelity rendering, human-computer interaction, multi-user, networked applications, perception, virtual reality

5 Real-time shading

◆ Marc Olano, Kurt Akeley, John C. Hart, Wolfgang Heidrich, Michael McCool, Jason L. Mitchell, Randi Rost
August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**

Publisher: ACM Press

Full text available:  pdf(7.39 MB) Additional Information: [full citation](#), [abstract](#)

Real-time procedural shading was once seen as a distant dream. When the first version of this course was offered four years ago, real-time shading was possible, but only with one-of-a-kind hardware or by combining the effects of tens to hundreds of rendering passes. Today, almost every new computer comes with graphics hardware capable of interactively executing shaders of thousands to tens of thousands of instructions. This course has been redesigned to address today's real-time shading capabili ...

6 Posters: Latency hiding through multithreading on a network processor

◆ Xiaofeng Guo, Jinquan Dai, Long Li, Zhiyuan Lv, Prashant R. Chandra
March 2007 **Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '07**

Publisher: ACM Press

Full text available:  pdf(321.71 KB) Additional Information: [full citation](#), [index terms](#)

Keywords: code motion, compiler, latency hiding, multicore

7 How to solve the current memory access and data transfer bottlenecks: at the processor architecture or at the compiler level

◆ Francky Catthoor, Nikil D. Dutt, Christoforos E. Kozyrakis
January 2000 **Proceedings of the conference on Design, automation and test in Europe DATE '00**

Publisher: ACM Press

Full text available:  pdf(92.68 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)
 Publisher Site

8 A dynamic optimization framework for a Java just-in-time compiler

◆ Toshio Suganuma, Toshiaki Yasue, Motohiro Kawahito, Hideaki Komatsu, Toshio Nakatani
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications OOPSLA '01**, Volume 36 Issue 11

Publisher: ACM Press

Full text available:  pdf(2.12 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The high performance implementation of Java Virtual Machines (JVM) and just-in-time (JIT) compilers is directed toward adaptive compilation optimizations on the basis of online runtime profile information. This paper describes the design and implementation of a dynamic optimization framework in a production-level Java JIT compiler. Our approach is to employ a mixed mode interpreter and a three level optimizing compiler, supporting quick, full, and special optimization, each of which has a differ ...

9 Balancing register allocation across threads for a multithreaded network processor

◆ Xiaotong Zhuang, Santosh Pande
June 2004 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2004 conference on Programming language design and implementation PLDI '04**, Volume 39 Issue 6

Publisher: ACM Press

Full text available:  pdf(429.85 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Modern network processors employ multi-threading to allow concurrency amongst multiple packet processing tasks. We studied the properties of applications running on the network processors and observed that their imbalanced register requirements across different threads at different program points could lead to poor performance. Many times application needs demand some threads to be more performance critical than others and thus by controlling the register allocation across threads one could impa ...

Keywords: multithreaded processor, network processor, register allocation

10 Multithreading I: Master/slave speculative parallelization

Craig Zilles, Gurindar Sohi
November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture MICRO 35**

Publisher: IEEE Computer Society Press

Full text available:  pdf(1.31 MB)  Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Master/Slave Speculative Parallelization (MSSP) is an execution paradigm for improving the execution rate of sequential programs by parallelizing them speculatively for execution on a multiprocessor. In MSSP, one processor---the master---executes an approximate version of the program to compute selected values that the full program's execution is expected to compute. The master's results are checked by slave processors that execute the original program. This validation is parallelized by cutting ...

11 Compilers: A framework for reducing instruction scheduling overhead in dynamic compilers

◆ Vikki Tang, Joran Siu, Alexander Vasilevskiy, Marcel Mitran
October 2006 **Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research CASCON '06**

Publisher: ACM Press

Full text available:  pdf(139.39 KB)

 htm(1.81 KB)

Additional Information: [full citation](#), [abstract](#), [references](#)

Start-up time is a serious concern for high-availability applications such as web servers, transaction managers, and batch processes. Compilation time contributes directly to start-up costs in dynamic compilers. Up to 30% of compilation time is spent scheduling

instructions in the IBM® Testarossa just-in-time compiler. In this paper, we describe a scheduling framework that reduces scheduling overhead by up to 61% with little to no degradation in throughput performance. By combining online pr ...

12 Evaluating titanium SPMD programs on the Tera MTA

◆ Carleton Miyamoto, Chang Lin

January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '99**

Publisher: ACM Press

Full text available:  [pdf\(314.08 KB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)



13 Array SSA form and its use in parallelization

◆ Kathleen Knobe, Vivek Sarkar

January 1998 **Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '98**

Publisher: ACM Press

Full text available:  [pdf\(1.99 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



14 Making Sequential Consistency Practical in Titanium

Amir Kamil, Jimmy Su, Katherine Yelick

November 2005 **Proceedings of the 2005 ACM/IEEE conference on Supercomputing SC '05**

Publisher: IEEE Computer Society

Full text available:  [pdf\(382.47 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

The memory consistency model in shared memory parallel programming controls the order in which memory operations performed by one thread may be observed by another. The most natural model for programmers is to have memory accesses appear to take effect in the order specified in the original program. Language designers have been reluctant to use this strong semantics, called sequential consistency, due to concerns over the performance of memory fence instructions and related mechanisms that guara ...



15 Object and native code thread mobility among heterogeneous computers (includes sources)

◆ B. Steensgaard, E. Jul

December 1995 **ACM SIGOPS Operating Systems Review , Proceedings of the fifteenth ACM symposium on Operating systems principles SOSP '95**, Volume 29 Issue 5

Publisher: ACM Press

Full text available:  [pdf\(1.50 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



16 An Application-Based Performance Characterization of the Columbia Supercluster

Rupak Biswas, M. Jahed Djomehri, Robert Hood, Haoqiang Jin, Cetin Kiris, Subhash Saini

November 2005 **Proceedings of the 2005 ACM/IEEE conference on Supercomputing SC '05**

Publisher: IEEE Computer Society

Full text available:  [pdf\(776.50 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)



Columbia is a 10,240-processor supercluster consisting of 20 Altix nodes with 512 processors each, and currently ranked as one of the fastest computers in the world. In this paper, we present the performance characteristics of Columbia obtained on up to four computing nodes interconnected via the InfiniBand and/or NUMAlink4 communication fabrics. We evaluate floatingpoint performance, memory bandwidth, message passing communication speeds, and compilers using a subset of the HPC Challenge benchm ...

Keywords: SGI Altix, multi-level parallelism, HPC Challenge benchmarks, NAS Parallel Benchmarks, molecular dynamics, multi-block overset grids, computational fluid dynamics

17 [Shangri-La: achieving high performance from compiled network applications while enabling ease of programming](#)

Michael K. Chen, Xiao Feng Li, Ruiqi Lian, Jason H. Lin, Lixia Liu, Tao Liu, Roy Ju
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6

Publisher: ACM Press

Full text available:  [pdf\(480.93 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Programming network processors is challenging. To sustain high line rates, network processors have extremely tight memory access and instruction budgets. Achieving desired performance has traditionally required hand-coded assembly. Researchers have recently proposed high-level programming languages for packet processing, but the challenges of compiling these languages into code that is competitive with hand-tuned assembly remain unanswered. This paper describes the Shangri-La compiler, which acce ...

Keywords: chip multiprocessors, dataflow programming, network processors, packet processing, program partitioning, throughput-oriented computing

18 [Session S6.2: compilers and program analysis: Experience with a retargetable compiler for a commercial network processor](#)

Jinhyun Kim, Sungjoon Jung, Yunheung Paek, Gang-Ryung Uh
October 2002 **Proceedings of the 2002 international conference on Compilers, architecture, and synthesis for embedded systems CASES '02**

Publisher: ACM Press

Full text available:  [pdf\(275.87 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The Paion PPII network processor is designed to meet the growing need for new high bandwidth network equipment. In order to rapidly reconfigure the processor for frequently varying internet services and technologies, a high performance compiler is urgently needed. Albeit various code generation techniques have been proposed for DSPs or ASIPs, we experienced these techniques are not easily tailored towards the target Paion PPII processor due to striking architectural differences. First, we will s ...

Keywords: compiler, network processor, non-orthogonal architecture

19 [The interactive performance of SLIM: a stateless, thin-client architecture](#)

Brian K. Schmidt, Monica S. Lam, J. Duane Northcutt
December 1999 **ACM SIGOPS Operating Systems Review , Proceedings of the seventeenth ACM symposium on Operating systems principles SOSP '99**, Volume 33 Issue 5

Publisher: ACM Press

Full text available:  [pdf\(1.79 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Taking the concept of thin clients to the limit, this paper proposes that desktop machines should just be simple, stateless I/O devices (display, keyboard, mouse, etc.) that access a shared pool of computational resources over a dedicated interconnection fabric --- much in the same way as a building's telephone services are accessed by a collection of handset devices. The stateless desktop design provides a useful mobility model in which users can transparently resume their work on any desktop c ...

20 [Compiler-directed channel allocation for saving power in on-chip networks](#)

Guangyu Chen, Feihui Li, Mahmut Kandemir
January 2006 **ACM SIGPLAN Notices , Conference record of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '06**, Volume 41 Issue 1

Publisher: ACM Press

Increasing complexity in the communication patterns of embedded applications parallelized over multiple processing units makes it difficult to continue using the traditional bus-based on-chip communication techniques. The main contribution of this paper is to demonstrate the importance of compiler technology in reducing power consumption of applications designed for emerging multi processor, NoC (Network-on-Chip) based embedded systems. Specifically, we propose and evaluate a compiler-directed a ...

Keywords: NoC, compiler, energy consumption

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



Terms used

[compiler optimization multi threading code motion critical section network processor](#)Found **60,664** of
199,787Sort results
by [Save results to a Binder](#)[Try an Advanced Search](#)Display
results [Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)

Results 21 - 40 of 200

Result page: [previous](#)[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale 

21 System-level power optimization: techniques and tools

 Luca Benini, Giovanni de Micheli April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**,

Volume 5 Issue 2

Publisher: ACM PressFull text available:  [pdf\(385.22 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic systems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, and storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survey ...

22 Compiler transformations for high-performance computing

 David F. Bacon, Susan L. Graham, Oliver J. Sharp December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4**Publisher:** ACM PressFull text available:  [pdf\(6.32 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

In the last three decades a large number of compiler transformations for optimizing programs have been implemented. Most optimizations for uniprocessors reduce the number of instructions executed by the program using transformations based on the analysis of scalar quantities and data-flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel processors maximize parallelism and memory locality with transformations that rely on tracking the properties o ...

Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism, superscalar processors, vectorization

23 Frontmatter (TOC, Letters, Election results, Software Reliability Resources!)

 Computing Curricula 2004 and the Software Engineering Volume SE2004, Software Reuse Research, ICSE 2005 Forward)July 2005 **ACM SIGSOFT Software Engineering Notes**, Volume 30 Issue 4**Publisher:** ACM PressFull text available:  [pdf\(6.19 MB\)](#) Additional Information: [full citation](#), [index terms](#)

24 Source-level global optimizations for fine-grain distributed shared memory systems

R. Veldema, R. F. H. Hofman, R. A. F. Bhoedjang, C. J. H. Jacobs, H. E. Bal



Publisher: ACM Press

Full text available:  pdf(112.60 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes and evaluates the use of aggressive static analysis in Jackal, a fine-grain Distributed Shared Memory (DSM) system for Java. Jackal uses an optimizing, source-level compiler rather than the binary rewriting techniques employed by most other fine-grain DSM systems. Source-level analysis makes existing access-check optimizations (e.g., access-check batching) more effective and enables two novel fine-grain DSM optimizations: object-graph aggregatio ...

25 Reducing NoC energy consumption through compiler-directed channel voltage



Guangyu Chen, Feihui Li, Mahmut Kandemir, Mary Jane Irwin

June 2006 ACM SIGPLAN Notices , Proceedings of the 2006 ACM SIGPLAN conference on Programming language design and implementation PLDI '06, Volume 41 Issue 6

Publisher: ACM Press

Full text available:  pdf(536.63 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

While scalable NoC (Network-on-Chip) based communication architectures have clear advantages over long point-to-point communication channels, their power consumption can be very high. In contrast to most of the existing hardware-based efforts on NoC power optimization, this paper proposes a compiler-directed approach where the compiler decides the appropriate voltage/frequency levels to be used for each communication channel in the NoC. Our approach builds and operates on a novel graph based rep ...

Keywords: compiler, energy, network-on-chip

26 Symbolic bounds analysis of pointers, array indices, and accessed memory regions



Radu Rugină, Martin C. Rinard

March 2005 ACM Transactions on Programming Languages and Systems (TOPLAS), Volume 27 Issue 2

Publisher: ACM Press

Full text available:  pdf(490.56 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This article presents a novel framework for the symbolic bounds analysis of pointers, array indices, and accessed memory regions. Our framework formulates each analysis problem as a system of inequality constraints between symbolic bound polynomials. It then reduces the constraint system to a linear program. The solution to the linear program provides symbolic lower and upper bounds for the values of pointer and array index variables and for the regions of memory that each statement and procedur ...

Keywords: Symbolic analysis, parallelization, static race detection

27 Pthreads for dynamic and irregular parallelism



Girija J. Narlikar, Guy E. Blelloch

November 1998 Proceedings of the 1998 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '98

Publisher: IEEE Computer Society

Full text available:  html(82.60 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

High performance applications on shared memory machines have typically been written in a coarse grained style, with one heavyweight thread per processor. In comparison, programming with a large number of lightweight, parallel threads has several advantages, including simpler coding for programs with irregular and dynamic parallelism, and better adaptability to a changing number of processors. The programmer can express a new thread to execute each individual parallel task; the implementation dyn ...

Keywords: Pthreads, dynamic scheduling, irregular parallelism, lightweight threads,

28 Coordinated parallelizing compiler optimizations and high-level synthesis

Sumit Gupta, Rajesh Kumar Gupta, Nikil D. Dutt, Alexandru Nicolau
October 2004 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 9 Issue 4

Publisher: ACM Press

Full text available:  pdf(923.65 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We present a high-level synthesis methodology that applies a coordinated set of coarse-grain and fine-grain parallelizing transformations. The transformations are applied both during a pre-synthesis phase and during scheduling, with the objective of optimizing the results of synthesis and reducing the impact of control flow constructs on the quality of results. We first apply a set of source level presynthesis transformations that include common sub-expression elimination (CSE), copy propagat ...

Keywords: Code motions, common subexpression elimination, dynamic CSE, embedded systems, high-level synthesis, parallelizing transformations, presynthesis

29 Critical path reduction for scalar programs

Michael Schlansker, Vinod Kathail

December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**

Publisher: IEEE Computer Society Press

Full text available:  pdf(1.38 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

30 Avoidance and suppression of compensation code in a trace scheduling compiler

Stefan M. Freudenberger, Thomas R. Gross, P. Geoffrey Lowney
July 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 16 Issue 4

Publisher: ACM Press

Full text available:  pdf(3.58 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Trace scheduling is an optimization technique that selects a sequence of basic blocks as a trace and schedules the operations from the trace together. If an operation is moved across basic block boundaries, one or more compensation copies may be required in the off-trace code. This article discusses the generation of compensation code in a trace scheduling compiler and presents techniques for limiting the amount of compensation code: avoidance (restricting code motion so that no compensatio ...

Keywords: SPEC89, instruction-level parallelism, performance evaluation, trace scheduling

31 Summary of ACM/ONR workshop on parallel and distributed debugging

January 1992 **ACM SIGOPS Operating Systems Review**, Volume 26 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.31 MB) Additional Information: [full citation](#), [citations](#), [index terms](#)

32 Mapping esterel onto a multi-threaded embedded processor

Xin Li, Marian Boldt, Reinhard von Hanxleden

October 2006 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , ACM SIGPLAN Notices , Proceedings of the 12th international conference on Architectural support for programming languages and operating systems ASPLOS-XII**, Volume 34 , 40 , 41 Issue 5 , 5 ,

The synchronous language Esterel is well-suited for programming control-dominated reactive systems at the system level. It provides non-traditional control structures, in particular concurrency and various forms of preemption, which allow to concisely express reactive behavior. As these control structures cannot be mapped easily onto traditional, sequential processors, an alternative approach that has emerged recently makes use of special-purpose reactive processors. However, the designs propose ...

Keywords: concurrency, esterel, low-power processing, multi-threading, reactive systems

33 Power reduction techniques for microprocessor systems

 Vasanth Venkatachalam, Michael Franz
September 2005 **ACM Computing Surveys (CSUR)**, Volume 37 Issue 3

Publisher: ACM Press

Full text available:  pdf(602.33 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Power consumption is a major factor that limits the performance of computers. We survey the "state of the art" in techniques that reduce the total power consumed by a microprocessor system over time. These techniques are applied at various levels ranging from circuits to architectures, architectures to system software, and system software to applications. They also include holistic approaches that will become more important over the next decade. We conclude that power management is a ...

Keywords: Energy dissipation, power reduction

34 Parallelizing load/stores on dual-bank memory embedded processors

 Xiaotong Zhuang, Santosh Pande
August 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 3

Publisher: ACM Press

Full text available:  pdf(746.64 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Many modern embedded processors such as DSPs support partitioned memory banks (also called X--Y memory or dual-bank memory) along with parallel load/store instructions to achieve higher code density and performance. In order to effectively utilize the parallel load/store instructions, the compiler must partition the memory-resident values and assign them to X or Y bank. This paper gives a postregister allocation solution to merge the generated load/store instructions into their parallel counterp ...

Keywords: DSP architectures, memory bank allocation, parallel load/stores, profile driven optimization

35 Computing curricula 2001

 September 2001 **Journal on Educational Resources in Computing (JERIC)**

Publisher: ACM Press

Full text available:  pdf(613.63 KB)  html(2.78 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

36 Data and memory optimization techniques for embedded systems

 P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P. G. Kjeldsberg
April 2001 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 6 Issue 2

Publisher: ACM Press

Full text available:  pdf(339.91 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a survey of the state-of-the-art techniques used in performing data and

memory-related optimizations in embedded systems. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three important cost metrics: area, performance, and power dissipation of the resulting implementation. We first examine architecture-independent optimizations in the form of code transformations. We next cover a broad spectrum of optimizati ...

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, data cache, data optimization, high-level synthesis, memory architecture customization, memory power dissipation, register file, size estimation, survey

37 "Flea-flicker" Multipass Pipelining: An Alternative to the High-Power Out-of-Order Offense

Ronald D. Barnes, Shane Ryoo, Wen-mei W. Hwu

November 2005 **Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture MICRO 38**

Publisher: IEEE Computer Society

Full text available:  pdf(346.82 KB)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

 Publisher Site

As microprocessor designs become increasingly powerand complexity-conscious, future microarchitectures must decrease their reliance on expensive dynamic scheduling structures. While compilers have generally proven adept at planning useful static instruction-level parallelism, relying solely on the compiler's instruction execution arrangement performs poorly when cache misses occur, because variable latency is not well tolerated. This paper proposes a new microarchitectural model, multipass pipel ...

38 Using high performance GIS software to visualize data: a hands-on software demonstration

Linda Burton, William Hatchett, Mari Hobkirk, Charles Powell

November 1998 **Proceedings of the 1998 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '98**

Publisher: IEEE Computer Society

Full text available:  html(80.49 KB) Additional Information: [full citation](#), [abstract](#), [references](#)

Since 1995 Wheat Ridge High School (WRHS) students have participated in a mapping project involving local open space, in conjunction with NASA. Students have learned to use *Idrisi*, a Geographical Imaging Systems (GIS) software, as well as other GIS programs *Arc View* and *Multispec*, to plan the location of a trail along Colorado's front range. As this project has progressed, students have learned the GIS technology as well as many science issues related to trail mapping. Simila ...

39 Compiler optimizations for power, performance: Architectural analysis and instruction-set optimization for design of network protocol processors

Haiyong Xie, Li Zhao, Laxmi Bhuyan

October 2003 **Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis CODES+ISSS '03**

Publisher: ACM Press

Full text available:  pdf(87.43 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

TCP/IP protocol processing latency has been an important issue in high-speed networks. In this paper, we present an architectural study of TCP/IP protocol. We port the TCP/IP protocol stack from the 4.4 FreeBSD to the SimpleScalar simulation environment. The architectural characteristics, such as instruction level parallelism and cache behavior, are studied through simulation. We also compare the characteristics of TCP/IP protocol to that of SPECint benchmark programs. It turns out that the form ...

Keywords: TCP/IP protocol, architecture simulation, instruction optimization, network processing

Publisher: ACM Press

Full text available:  pdf(1.72 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The Connection Machine® computer system supports a data parallel programming style, making it a natural target architecture for Fortran 8x array constructs. The Connection Machine Fortran compiler generates VAX code that performs scalar operations and directs the Connection Machine to perform array operations. The Connection Machine virtual processor mechanism supports elemental operations on very large arrays. Most array operators and intrinsic functions map into single instructions or ...

Results 21 - 40 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



Terms used

[compiler optimization](#) [multi threading](#) [code motion](#) [critical section](#) [network processor](#)Found **60,664** of
199,787Sort results
by relevance Save results to a Binder[Try an Advanced Search](#)Display
results expanded form Search Tips[Try this search in The ACM Guide](#) Open results in a new
windowResults 41 - 60 of 200 Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale

**41 Optimizing parallel programs with explicit synchronization**

Arvind Krishnamurthy, Katherine Yelick

 June 1995 **ACM SIGPLAN Notices**, Proceedings of the ACM SIGPLAN 1995 conference on Programming language design and implementation PLDI '95, Volume 30

Issue 6

Publisher: ACM PressFull text available: [pdf\(1.07 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present compiler analyses and optimizations for explicitly parallel programs that communicate through a shared address space. Any type of code motion on explicitly parallel programs requires a new kind of analysis to ensure that operations reordered on one processor cannot be observed by another. The analysis, based on work by Shasha and Snir, checks for cycles among interfering accesses. We improve the accuracy of their analysis by using additional information from post-wait synchroniza ...

42 1 - Special Section: Link-time compaction and optimization of ARM executables

Bjorn De Sutter, Ludo Van Put, Dominique Chanet, Bruno De Bus, Koen De Bosschere

 February 2007 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 6

Issue 1

Publisher: ACM PressFull text available: [pdf\(636.53 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The overhead in terms of code size, power consumption, and execution time caused by the use of precompiled libraries and separate compilation is often unacceptable in the embedded world, where real-time constraints, battery life-time, and production costs are of critical importance. In this paper, we present our link-time optimizer for the ARM architecture. We discuss how we can deal with the peculiarities of the ARM architecture related to its visible program counter and how the introduced over ...

Keywords: Performance, compaction, linker, optimization**43 A lifetime optimal algorithm for speculative PRE**

Jingling Xue, Qiong Cai

 June 2006 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 3

Issue 2

Publisher: ACM PressFull text available: [pdf\(1.38 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

A lifetime optimal algorithm, called MC-PRE, is presented for the first time that performs speculative PRE based on edge profiles. In addition to being computationally optimal in the sense that the total number of dynamic computations for an expression in the transformed code is minimized, MC-PRE is also lifetime optimal since the lifetimes of introduced temporaries are also minimized. The key in achieving lifetime optimality lies not only in finding a unique minimum cut on a transformed graph o ...

Keywords: Partial redundancy elimination, classic PRE, computational optimality, data-flow analysis, lifetime optimality, speculative PRE

44 The program decision logic approach to predicated execution 

David I. August, John W. Sias, Jean-Michel Puiatti, Scott A. Mahlke, Daniel A. Connors, Kevin M. Crozier, Wen-mei W. Hwu

May 1999 **ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture ISCA '99**, Volume 27 Issue 2

Publisher: IEEE Computer Society, ACM Press

Full text available:  pdf(215.39 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index](#)

 Publisher Site

terms

Modern compilers must expose sufficient amounts of Instruction-Level Parallelism (ILP) to achieve the promised performance increases of superscalar and VLIW processors. One of the major impediments to achieving this goal has been inefficient programmatic control flow. Historically, the compiler has translated the programmer's original control structure directly into assembly code with conditional branch instructions. Eliminating inefficiencies in handling branch instructions and exploiting ILP h ...

45 Architectures and performance analysis: New approach to architectural synthesis: 

◆ **incorporating QoS constraint**

Harsh Dhand, Basant Dwivedi, M Balakrishnan

October 2006 **Proceedings of the 6th ACM & IEEE International conference on Embedded software EMSOFT '06**

Publisher: ACM Press

Full text available:  pdf(209.86 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index](#) terms

Embedded applications like video decoding, video streaming and those in the network domain, typically have a Quality of Service (QoS) requirement which needs to be met. Apart from being a design constraint, it can also be considered as a flexibility that the design does not have to work under worst case data condition. For example, in the case of video decoding with variable decoding time for individual frames, it may be adequate that only a fraction of frames (say 90%) needs to be decoded. In t ...

Keywords: mapping, partitioning, process network, quality of service, soft real time constraints

46 Optimizing for space and time usage with speculative partial redundancy elimination 

◆ Bernhard Scholz, Nigel Horspool, Jens Knoop

June 2004 **ACM SIGPLAN Notices , Proceedings of the 2004 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems LCTES '04**, Volume 39 Issue 7

Publisher: ACM Press

Full text available:  pdf(177.57 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index](#) terms

Speculative partial redundancy elimination (SPRE) uses execution profiles to improve the expected performance of programs. We show how the problem of placing expressions to achieve the optimal expected performance can be mapped to a particular kind of network flow problem and hence solved by well known techniques. Our solution is sufficiently efficient to be used in practice. Furthermore, the objective function may be chosen so that reduction in space requirements is the primary goal and executi ...

Keywords: code motion, common subexpressions, partial redundancy, profile-guided optimization, speculation

47 Multithreading and multiprocessing: A case study of multi-threading in the embedded 

◆ **space**

Greg Hoover, Forrest Brewer, Timothy Sherwood

The continuing miniaturization of technology coupled with wireless networks has made it feasible to physically embed sensor network systems into the environment. Sensor net processors are tasked with the job of handling a disparate set of interrupt driven activity, from networks to timers to the sensors themselves. In this paper, we demonstrate the advantages of a tiny multi-threaded microcontroller design which targets embedded applications that need to respond to events at high speed. While mult ...

Keywords: embedded architecture, multi-threading

48 [Emerging areas: Programming challenges in network processor deployment](#)



Chidamber Kulkarni, Matthias Gries, Christian Sauer, Kurt Keutzer

◆ October 2003 **Proceedings of the 2003 international conference on Compilers,
architecture and synthesis for embedded systems CASES '03**

Publisher: ACM Press

Full text available:  pdf(234.71 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Programming multi-processor ASIPs, such as network processors, remains an art due to the wide variety of architectures and due to little support for exploring different implementation alternatives. We present a study that implements an IP forwarding router application on two different network processors to better understand the main challenges in programming such multi-processor ASIPs. The goal of this study is to identify the elements central to a successful deployment of such systems based on ...

Keywords: IPv4 forwarding, mapping, multi-threading, programming heterogeneous architectures, programming model, resource sharing

49 [Taming the IXP network processor](#)



Lal George, Matthias Blume

◆ May 2003 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2003 conference
on Programming language design and implementation PLDI '03**, Volume 38

Issue 5

Publisher: ACM Press

Full text available:  pdf(159.27 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

We compile Nova, a new language designed for writing network processing applications, using a back end based on integer-linear programming (ILP) for register allocation, optimal bank assignment, and spills. The compiler's optimizer employs CPS as its intermediate representation; some of the invariants that this IR guarantees are essential for the formulation of a practical ILP model. Appel and George used a similar ILP-based technique for the IA32 to decide which variables reside in registers but ...

Keywords: Intel IXA, bank assignment, code generation, integer linear programming, network processors, programming languages, register allocation

50 [Tuning compiler optimizations for simultaneous multithreading](#)



Jack L. Lo, Susan J. Eggers, Henry M. Levy, Sujay S. Parekh, Dean M. Tullsen

December 1997 **Proceedings of the 30th annual ACM/IEEE international symposium
on Microarchitecture MICRO 30**

Publisher: IEEE Computer Society

Full text available:  pdf(1.45 MB)  Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)
[Publisher Site](#)

Compiler optimizations are often driven by specific assumptions about the underlying architecture and implementation of the target machine. For example, when targeting shared-memory multiprocessors, parallel programs are compiled to minimize sharing, in

order to decrease high-cost, inter-processor communication. This paper reexamines several compiler optimizations in the context of simultaneous multithreading (SMT), a processor architecture that issues instructions from multiple threads to the f ...

Keywords: cache size, compiler optimizations, cyclic algorithm, fine-grained sharing, instructions, inter-processor communication, inter-thread instruction-level parallelism, latency hiding, loop tiling, loop-iteration scheduling, memory system resources, optimising compilers, parallel architecture, parallel programs, performance, processor architecture, shared-memory multiprocessors, simultaneous multithreading, software speculative execution

51 Supercomputers: Evaluating support for global address space languages on the Cray X1

Christian Bell, Wei-Yu Chen, Dan Bonachea, Katherine Yelick

June 2004 **Proceedings of the 18th annual international conference on Supercomputing ICS '04**

Publisher: ACM Press

Full text available:  [pdf\(265.56 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The Cray X1 was recently introduced as the first in a new line of parallel systems to combine high-bandwidth vector processing with an MPP system architecture. Alongside capabilities such as automatic fine-grained data parallelism through the use of vector instructions, the X1 offers hardware support for a transparent global-address space (GAS), which makes it an interesting target for GAS languages. In this paper, we describe our experience with developing a portable, open-source and high perfo ...

Keywords: UPC, X1, global address space

52 Compilation: Efficient spill code for SDRAM

V. Krishna Nandivada, Jens Palsberg

October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03**

Publisher: ACM Press

Full text available:  [pdf\(199.32 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Processors such as StrongARM and memory such as SDRAM enable efficient execution of multiple loads and stores in a single instruction. This is particularly useful in connection with register allocation where spill code may need to save and restore multiple registers. Until now, there has been no effective strategy for utilizing this to its full potential. In this paper we investigate the use of SDRAM for optimization of spill code. The core of the problem is to arrange the variables in the spill ...

Keywords: SDRAM, integer linear programming, memory layout, optimization

53 Software thread integration for embedded system display applications

Alexander G. Dean

February 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 1

Publisher: ACM Press

Full text available:  [pdf\(1.40 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Embedded systems require control of many concurrent real-time activities, leading to system designs that feature a variety of hardware peripherals, with each providing a specific, dedicated service. These peripherals increase system size, cost, weight, and design time. Software thread integration (STI) provides low-cost thread concurrency on general-purpose processors by automatically interleaving multiple threads of control into one. This simplifies hardware to software migration (which elimina ...

Keywords: fine-grain concurrency, hardware to software migration, software thread integration

54 Improving the performance of speculatively parallel applications on the Hydra CMP

◆ Kunle Olukotun, Lance Hammond, Mark Willey
May 1999 **Proceedings of the 13th international conference on Supercomputing ICS '99**

Publisher: ACM Press

Full text available:  pdf(1.66 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



Keywords: chip multiprocessor, data speculation, feedback-driven optimization, multithreading, parallel programming, performance evaluation



55 HPFBench: a high performance Fortran benchmark suite

◆ Y. Charlie Hu, Guohua Jin, S. Lennart Johnsson, Dimitris Kehagias, Nadia Shalaby
March 2000 **ACM Transactions on Mathematical Software (TOMS)**, Volume 26 Issue 1

Publisher: ACM Press

Full text available:  pdf(274.52 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The high performance Fortran (HPF) benchmark suite HPFBench is designed for evaluating the HPF language and compilers on scalable architectures. The functionality of the benchmarks covers scientific software library functions and application kernels that reflect the computational structure and communication patterns in fluid dynamic simulations, fundamental physics, and molecular studies in chemistry and biology. The benchmarks are characterized in terms of FLOP count, memory usage, communi ...

Keywords: benchmarks, compilers, high performance Fortran



56 Special issue: AI in engineering

◆ D. Sriram, R. Joobani
April 1985 **ACM SIGART Bulletin**, Issue 92

Publisher: ACM Press

Full text available:  pdf(8.79 MB) Additional Information: [full citation](#), [abstract](#)

The papers in this special issue were compiled from responses to the announcement in the July 1984 issue of the SIGART newsletter and notices posted over the ARPAnet. The interest being shown in this area is reflected in the sixty papers received from over six countries. About half the papers were received over the computer network.



57 Meta optimization: improving compiler heuristics with machine learning

◆ Mark Stephenson, Saman Amarasinghe, Martin Martin, Una-May O'Reilly
May 2003 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation PLDI '03**, Volume 38 Issue 5

Publisher: ACM Press

Full text available:  pdf(302.23 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Compiler writers have crafted many heuristics over the years to approximately solve NP-hard problems efficiently. Finding a heuristic that performs well on a broad range of applications is a tedious and difficult process. This paper introduces Meta Optimization, a methodology for automatically fine-tuning compiler heuristics. Meta Optimization uses machine-learning techniques to automatically search the space of compiler heuristics. Our techniques reduce compiler design complexity by relieving c ...

Keywords: compiler heuristics, genetic programming, machine learning, priority functions



58 On the validity of trace-driven simulation for multiprocessors

Publisher: ACM Press

Full text available:  pdf(840.99 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

59 Fast and efficient searches for effective optimization-phase sequences 

◆ Prasad A. Kulkarni, Stephen R. Hines, David B. Whalley, Jason D. Hiser, Jack W. Davidson,

Douglas L. Jones

June 2005 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 2

Issue 2

Publisher: ACM Press

Full text available:  pdf(1.69 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

It has long been known that a fixed ordering of optimization phases will not produce the best code for every application. One approach for addressing this phase-ordering problem is to use an evolutionary algorithm to search for a specific sequence of phases for each module or function. While such searches have been shown to produce more efficient code, the approach can be extremely slow because the application is compiled and possibly executed to evaluate each sequence's effectiveness. Consequen ...

Keywords: Phase ordering, genetic algorithms, interactive compilation

60 Scheduling constrained dynamic applications on clusters 

◆ Kathleen Knobe, James M. Rehg, Arun Chauhan, Rishiyur S. Nikhil, Umakishore

Ramachandran

January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '99**

Publisher: ACM Press

Full text available:  pdf(189.17 KB) Additional Information: [full citation](#), [references](#), [index terms](#)

Results 41 - 60 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Terms used

[compiler optimization](#) [multi threading](#) [code motion](#) [critical section](#) [network processor](#)Found **60,664** of
199,787Sort results
by [Save results to a Binder](#)[Try an Advanced Search](#)Display
results [Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)

Results 61 - 80 of 200

Result page: [previous](#)[1](#) [2](#) [3](#) **4** [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale **61 Efficient and correct execution of parallel programs that share memory** Dennis Shasha, Marc Snir April 1988 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 10 Issue 2

Publisher: ACM PressFull text available:  [pdf\(2.35 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

In this paper we consider an optimization problem that arises in the execution of parallel programs on shared-memory multiple-instruction-stream, multiple-data-stream (MIMD) computers. A program on such machines consists of many sequential program segments, each executed by a single processor. These segments interact as they access shared variables. Access to memory is asynchronous, and memory accesses are not necessarily executed in the order they were issued. An execution is correct if it ...

62 Multigrain shared memory Donald Yeung, John Kubiatowicz, Anant Agarwal May 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 2**Publisher:** ACM PressFull text available:  [pdf\(369.18 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Parallel workstations, each comprising tens of processors based on shared memory, promise cost-effective scalable multiprocesssing. This article explores the coupling of such small- to medium-scale shared-memory multiprocessors through software over a local area network to synthesize larger shared-memory systems. We call these systems Distributed Shared-memory MultiProcessors (DSMPs). This article introduces the design of a shared-memory system that uses multiple granularities of sharing, ca ...

Keywords: distributed memory, symmetric multiprocessors, system of systems**63 Compiler and runtime support for efficient software transactional memory** Ali-Reza Adl-Tabatabai, Brian T. Lewis, Vijay Menon, Brian R. Murphy, Bratin Saha, Tatiana Shpeisman June 2006 **ACM SIGPLAN Notices , Proceedings of the 2006 ACM SIGPLAN conference on Programming language design and implementation PLDI '06**, Volume 41

Issue 6

Publisher: ACM PressFull text available:  [pdf\(211.55 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Programmers have traditionally used locks to synchronize concurrent access to shared data. Lock-based synchronization, however, has well-known pitfalls: using locks for fine-grain synchronization and composing code that already uses locks are both difficult and prone to deadlock. Transactional memory provides an alternate concurrency control

mechanism that avoids these pitfalls and significantly eases concurrent programming. Transactional memory language constructs have recently been proposed as ...

Keywords: code generation, compiler optimizations, locking, synchronization, transactional memory, virtual machines

64 Instruction fetch and control flow: Reducing control overhead in dataflow architectures



Andrew Petersen, Andrew Putnam, Martha Mercaldi, Andrew Schwerin, Susan Eggers, Steve Swanson, Mark Oskin

September 2006 **Proceedings of the 15th international conference on Parallel architectures and compilation techniques PACT '06**

Publisher: ACM Press

Full text available: [pdf\(662.72 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

In recent years, computer architects have proposed tiled architectures in response to several emerging problems in processor design, such as design complexity, wire delay, and fabrication reliability. One of these architectures, WaveScalar, uses a dynamic, tagged-token dataflow execution model to simplify the design of the processor tiles and their interconnection network and to achieve good parallel performance. However, using a dataflow execution model reawakens old problems, including the ins ...

Keywords: Wavescalar, compiler, dataflow, tiled architecture

65 Support for High-Frequency Streaming in CMPs



Ram Rangan, Neil Vachharajani, Adam Stoler, Guilherme Ottoni, David I. August, George Z. N. Cai

December 2006 **Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture MICRO 39**

Publisher: IEEE Computer Society

Full text available: [pdf\(283.43 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

As the industry moves toward larger-scale chip multiprocessors, the need to parallelize applications grows. High inter-thread communication delays, exacerbated by overstressed high-latency memory subsystems and ever-increasing wire delays, require parallelization techniques to create partially or fully independent threads to improve performance. Unfortunately, developers and compilers alike often fail to find sufficient independent work of this kind. Recently proposed pipelined streaming techni ...

66 Measurement and evaluation of the MIPS architecture and processor



Thomas R. Gross, John L. Hennessy, Steven A. Przybylski, Christopher Rowen

August 1988 **ACM Transactions on Computer Systems (TOCS)**, Volume 6 Issue 3

Publisher: ACM Press

Full text available: [pdf\(2.30 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

MIPS is a 32-bit processor architecture that has been implemented as an nMOS VLSI chip. The instruction set architecture is RISC-based. Close coupling with compilers and efficient use of the instruction set by compiled programs were goals of the architecture. The MIPS architecture requires that the software implement some constraints in the design that are normally considered part of the hardware implementation. This paper presents experimental results on the effectiveness of this processor ...

67 Technical reports



SIGACT News Staff

January 1980 **ACM SIGACT News**, Volume 12 Issue 1

Publisher: ACM Press

Full text available: [pdf\(5.28 MB\)](#) Additional Information: [full citation](#)

The rapid growth of device densities on silicon has made it feasible to deploy reconfigurable hardware as a highly parallel computing platform. However, one of the obstacles to the wider acceptance of this technology is its programmability. The application needs to be programmed in hardware description languages or an assembly equivalent, whereas most application programmers are used to the algorithmic programming paradigm. SA-C has been proposed as an expression-oriented language designed to im ...

Keywords: Reconfigurable computing, SIMD, compilers

69 VISTA: VPO interactive system for tuning applications
◆ Prasad Kulkarni, Wankang Zhao, Stephen Hines, David Whalley, Xin Yuan, Robert van Engelen, Kyle Gallivan, Jason Hiser, Jack Davidson, Baosheng Cai, Mark Bailey, Hwashin Moon, Kyunghwan Cho, Yunheung Paek
November 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5
Issue 4
Publisher: ACM Press
Full text available: [pdf\(4.01 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Software designers face many challenges when developing applications for embedded systems. One major challenge is meeting the conflicting constraints of speed, code size, and power consumption. Embedded application developers often resort to hand-coded assembly language to meet these constraints since traditional optimizing compiler technology is usually of little help in addressing this challenge. The results are software systems that are not portable, less robust, and more costly to develop an ...

Keywords: User-directed code improvement, genetic algorithms, interactive compilation, phase ordering

70 Compilation: Cluster assignment of global values for clustered VLIW processors
◆ Andrei Terechko, Erwan Le Thénaff, Henk Corporaal
October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03**
Publisher: ACM Press
Full text available: [pdf\(330.94 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

In this paper high-level language (HLL) variables that are alive in a whole HLL function, across multiple scheduling units, are termed as global values. Due to their long live ranges and, hence, large impact on the schedule, the global values require different compiler optimizations than local values, which span across only one scheduling unit. The instruction scheduler for a clustered ILP processor, which is responsible for cluster assignment of operations and variables, faces a difficult prob ...

Keywords: ILP, VLIW, cluster assignment, compiler, instruction scheduler, register allocation

71 Helper threads via virtual multithreading on an experimental itanium® 2 processor-based platform
◆ Perry H. Wang, Jamison D. Collins, Hong Wang, Dongkeun Kim, Bill Greene, Kai-Ming Chan, Aamir B. Yunus, Terry Sych, Stephen F. Moore, John P. Shen
October 2004 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the 11th international conference on Architectural support for programming**

Publisher: ACM Press

Full text available:  pdf(225.47 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Helper threading is a technology to accelerate a program by exploiting a processor's multithreading capability to run ``assist'' threads. Previous experiments on hyper-threaded processors have demonstrated significant speedups by using helper threads to prefetch hard-to-predict delinquent data accesses. In order to apply this technique to processors that do not have built-in hardware support for multithreading, we introduce virtual multithreading (VMT), a novel form of switch-on-event user-level ...

Keywords: DB2 database, PAL, cache miss prefetching, helper thread, itanium processor, multithreading, switch-on-event

72 A study of devirtualization techniques for a Java Just-In-Time compiler 

 Kazuaki Ishizaki, Motohiro Kawahito, Toshiaki Yasue, Hideaki Komatsu, Toshio Nakatani
October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '00**, Volume 35 Issue 10

Publisher: ACM Press

Full text available:  pdf(225.89 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Many devirtualization techniques have been proposed to reduce the runtime overhead of dynamic method calls for various object-oriented languages, however, most of them are less effective or cannot be applied for Java in a straightforward manner. This is partly because Java is a statically-typed language and thus transforming a dynamic call to a static one does not make a tangible performance gain (owing to the low overhead of accessing the method table) unless it is inlined, and partly because t ...

73 Code and data layout optimisations for embedded software: An interprocedural code optimization technique for network processors using hardware multi-threading support 

Hanno Scharwaechter, Manuel Hohenauer, Rainer Leupers, Gerd Ascheid, Heinrich Meyr
March 2006 **Proceedings of the conference on Design, automation and test in Europe: Proceedings DATE '06**

Publisher: European Design and Automation Association

Full text available:  pdf(174.69 KB) Additional Information: [full citation](#), [abstract](#), [references](#)

Sophisticated C compiler support for network processors (NPUs) is required to improve their usability and consequently, their acceptance in system design. Nonetheless, high-level code compilation always introduces overhead, regarding code size and performance compared to handwritten assembly code. This overhead results partially from high-level function calls that usually introduce memory accesses in order to save and reload register contents. A key feature of many NPU architectures is hardware ...

74 A generational mostly-concurrent garbage collector 

 Tony Printezis, David Detlefs
October 2000 **ACM SIGPLAN Notices , Proceedings of the 2nd international symposium on Memory management ISMM '00**, Volume 36 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.67 MB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

This paper reports our experiences with a mostly-concurrent incremental garbage collector, implemented in the context of a high performance virtual machine for the Java™ programming language. The garbage collector is based on the ``mostly parallel'' collection algorithm of Boehm et al. and can be used as the old generation of a generational memory system. It overloads efficient write-barrier code already generated to support generational garbage collection to also ident ...

75 Exploiting coarse-grained task, data, and pipeline parallelism in stream programs 

Michael I. Gordon, William Thies, Saman Amarasinghe

Publisher: ACM Press

Full text available:  pdf(516.79 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

As multicore architectures enter the mainstream, there is a pressing demand for high-level programming models that can effectively map to them. Stream programming offers an attractive way to expose coarse-grained parallelism, as streaming applications (image, video, DSP, etc.) are naturally represented by independent filters that communicate over explicit data channels. In this paper, we demonstrate an end-to-end stream compiler that attains robust multicore performance in the face of varying app ...

Keywords: Raw, StreamIt, coarse-grained dataflow, multicore, software pipelining, streams

76 Compilers I: A performance analysis of the Berkeley UPC compiler 

Parry Husbands, Costin Iancu, Katherine Yelick

June 2003 **Proceedings of the 17th annual international conference on Supercomputing ICS '03**

Publisher: ACM Press

Full text available:  pdf(137.75 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Unified Parallel C (UPC) is a parallel language that uses a Single Program Multiple Data (SPMD) model of parallelism within a global address space. The global address space is used to simplify programming, especially on applications with irregular data structures that lead to fine-grained sharing between threads. Recent results have shown that the performance of UPC using a commercial compiler is comparable to that of MPI [7]. In this paper we describe a portable open source compiler for UPC. Ou ...

Keywords: UPC, global address space, parallel, performance

77 Lightweight Implementation of the POSIX Threads API for an On-Chip MIPS Multiprocessor with VCI Interconnect 

Frederic Petrot, Pascal Gomez

March 2003 **Proceedings of the conference on Design, Automation and Test in Europe: Designers' Forum - Volume 2 DATE '03**

Publisher: IEEE Computer Society

Full text available:  pdf(132.95 KB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)
 Publisher Site

This paper relates our experience in designing from scratch a multi-threaded kernel for a MIPS R3000 on-chip multiprocessor. We briefly present the target architecture build around a VCI compliant interconnect, and the CPU characteristics. Then we focus on the implementation of part of the POSIX 1003.1b and 1003.1c standards. We conclude this case study by simulation results obtained by cycle true simulation of an MJPEG video decoder application on the multiprocessor, using several scheduler org ...

78 A Performance Evaluation of the Convex SPP-1000 Scalable Shared Memory Parallel Computer 

Thomas Sterling, Daniel Savarese, Peter MacNeice, Kevin Olson, Clark Mobarry, Bruce Fryxell, Phillip Merkey

December 1995 **Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM) - Volume 00 Supercomputing '95**

Publisher: ACM Press

Full text available:  pdf(457.11 KB)  html(2.67 KB) Additional Information: [full citation](#), [references](#), [citations](#)
 ps(780.23 KB)

79 Runtime optimizations for a Java DSM implementation

R. Veldema, R. F. H. Hofman, R. A. F. Bhoedjang, H. E. Bal

June 2001 **Proceedings of the 2001 joint ACM-ISCOPE conference on Java Grande JGI '01**

Publisher: ACM Press

Full text available:  [pdf\(740.71 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Jackal is a fine-grained distributed shared memory implementation of the Java programming language. Jackal implements Java's memory model and allows multithreaded Java programs to run unmodified on distributed-memory systems.

This paper focuses on Jackal's runtime system, which implements a multiple-writer, home-based consistency protocol. Protocol actions are triggered by software access checks that Jackal's compiler inserts before object and array references. We describe optimizatio ...

80 Fast searches for effective optimization phase sequences

Prasad Kulkarni, Stephen Hines, Jason Hiser, David Whalley, Jack Davidson, Douglas Jones

June 2004 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2004 conference on Programming language design and implementation PLDI '04**, Volume 39

Issue 6

Publisher: ACM Press

Full text available:  [pdf\(862.40 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

It has long been known that a fixed ordering of optimization phases will not produce the best code for every application. One approach for addressing this phase ordering problem is to use an evolutionary algorithm to search for a specific sequence of phases for each module or function. While such searches have been shown to produce more efficient code, the approach can be extremely slow because the application is compiled and executed to evaluate each sequence's effectiveness. Consequently, evol ...

Keywords: genetic algorithms, interactive compilation, phase ordering

Results 61 - 80 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) **4** [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Terms used

compiler optimization multi threading code motion critical section network processor

Found **60,664** of
199,787

Sort results
by

[Save results to a Binder](#)

Try an [Advanced Search](#)

Display
results

[Search Tips](#)

Try this search in [The ACM Guide](#)

[Open results in a new window](#)

Results 81 - 100 of 200 Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) **5** [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale 

81 Communication optimizations for parallel C programs

 Yingchun Zhu, Laurie J. Hendren

 May 1998 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1998 conference on Programming language design and implementation PLDI '98**, Volume 33

Issue 5

Publisher: ACM Press

Full text available:  [pdf\(1.76 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents algorithms for reducing the communication overhead for parallel C programs that use dynamically-allocated data structures. The framework consists of an analysis phase called *possible-placement analysis*, and a transformation phase called *communication selection*. The fundamental idea of *possible-placement analysis* is to find all possible points for insertion of remote memory operations. Remote reads are propagated upwards, whereas remote writes are propagate ...

82 Enhanced code compression for embedded RISC processors

 Keith D. Cooper, Nathaniel McIntosh

 May 1999 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1999 conference on Programming language design and implementation PLDI '99**, Volume 34

Issue 5

Publisher: ACM Press

Full text available:  [pdf\(1.31 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper explores compiler techniques for reducing the memory needed to load and run program executables. In embedded systems, where economic incentives to reduce both RAM and ROM are strong, the size of compiled code is increasingly important. Similarly, in mobile and network computing, the need to transmit an executable before running it places a premium on code size. Our work focuses on reducing the size of a program's code segment, using pattern-matching techniques to identify and coalesce ...

83 The elements of nature: interactive and realistic techniques

 Oliver Deussen, David S. Ebert, Ron Fedkiw, F. Kenton Musgrave, Przemyslaw Prusinkiewicz, Doug Roble, Jos Stam, Jerry Tessendorf

August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**

Publisher: ACM Press

Full text available:  [pdf\(17.65 MB\)](#) Additional Information: [full citation](#), [abstract](#)

This updated course on simulating natural phenomena will cover the latest research and production techniques for simulating most of the elements of nature. The presenters will provide movie production, interactive simulation, and research perspectives on the difficult task of photorealistic modeling, rendering, and animation of natural phenomena. The course offers a nice balance of the latest interactive graphics hardware-based simulation techniques and the latest physics-based simulation techni ...

84 Context-specific middleware specialization techniques for optimizing software product-line architectures



Arvind S. Krishna, Aniruddha S. Gokhale, Douglas C. Schmidt

April 2006 **ACM SIGOPS Operating Systems Review , Proceedings of the 2006**

EuroSys conference EuroSys '06, Volume 40 Issue 4

Publisher: ACM Press

Full text available: [pdf\(676.55 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Product-line architectures (PLAs) are an emerging paradigm for developing software families for distributed real-time and embedded (DRE) systems by customizing reusable artifacts, rather than hand-crafting software from scratch. To reduce the effort of developing software PLAs and product variants for DRE systems, developers are applying general-purpose -- ideally standard -- middleware platforms whose reusable services and mechanisms support a range of application quality of service (QoS) requi ...

Keywords: middleware, product lines, specializations

85 Neural networks and dynamic complex systems



Geoffrey Fox, Wojtek Furmanski, Alex Ho, Jeff Koller, Peter Simic, Isaac Wong

March 1989 **Proceedings of the 22nd annual symposium on Simulation ANSS '89**

Publisher: IEEE Computer Society Press

Full text available: [pdf\(1.44 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We describe the use of neural networks for optimization and inference associated with a variety of complex systems. We show how a string formalism can be used for parallel computer decomposition, message routing and sequential optimizing compilers. We extend these ideas to a general treatment of spatial assessment and distributed artificial intelligence.

86 Intraprogram dynamic voltage scaling: Bounding opportunities with analytic modeling



Fen Xie, Margaret Martonosi, Sharad Malik

September 2004 **ACM Transactions on Architecture and Code Optimization (TACO)**,

Volume 1 Issue 3

Publisher: ACM Press

Full text available: [pdf\(980.11 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Dynamic voltage scaling (DVS) has become an important dynamic power-management technique to save energy. DVS tunes the power-performance tradeoff to the needs of the application. The goal is to minimize energy consumption while meeting performance needs. Since CPU power consumption is strongly dependent on the supply voltage, DVS exploits the ability to control the power consumption by varying a processor's supply voltage and clock frequency. However, because of the energy and time overhead asso ...

Keywords: Analytical model, compiler, dynamic voltage scaling, low power, mixed-integer linear programming

87 Tuning In-Sensor Data Filtering to Reduce Energy Consumption in Wireless Sensor Networks



I. Kadıyif, M. Kandemir

February 2004 **Proceedings of the conference on Design, automation and test in Europe - Volume 2 DATE '04**

Publisher: IEEE Computer Society

Full text available: [pdf\(136.20 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

In recent years, research on wireless sensor networks has been undergoing a revolution, promising to have significant impact on a broad range of applications from military to health care to food safety. An important problem in many sensor network applications is to decide the amount of computation (or filtering) that needs to be done in the sensor nodes before the data are shifted to a central base station. Right amount of data filtering in the sensor nodes can lead to large savings in network-w ...

88 **Technical papers: MPI and communication---Software routing and aggregation of messages to optimize the performance of HPCC randomaccess benchmark**
Rahul Garg, Yogish Sabharwal
November 2006 **Proceedings of the 2006 ACM/IEEE conference on Supercomputing SC '06**
Publisher: ACM Press
Full text available: [pdf\(158.36 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)
 [html\(2.02 KB\)](#)

The HPC Challenge(HPCC) benchmark suite is increasingly being used to evaluate the performance of supercomputers. It augments the traditional LINPACK benchmark by adding six more benchmarks, each designed to measure a specific aspect of the system performance. In this paper, we analyze the HPCC Randomaccess benchmark which is designed to measure the performance of random memory updates. We show that, on many systems, the bisection bandwidth of the network may be the performance bottleneck of this ...

Keywords: Blue Gene/L, HPC challenge, benchmarks, high performance computing, linpack, randomaccess, supercomputing

89 **Embedded applications: Architectural optimizations for low-power, real-time speech recognition**
 Rajeev Krishna, Scott Mahlke, Todd Austin
October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03**
Publisher: ACM Press
Full text available: [pdf\(634.03 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The proliferation of computing technology to low power domains such as hand--held devices has lead to increased interest in portable interface technologies, with particular interest in speech recognition. The computational demands of robust, large vocabulary speech recognition systems, however, are currently prohibitive for such low power devices. This work begins an exploration of domain specific characteristics of speech recognition that might be exploited to achieve the requisite performance w ...

90 **APRIL: a processor architecture for multiprocessing**
 Anant Agarwal, Beng-Hong Lim, David Kranz, John Kubiatowicz
May 1990 **ACM SIGARCH Computer Architecture News , Proceedings of the 17th annual international symposium on Computer Architecture ISCA '90**, Volume 18 Issue 3a
Publisher: ACM Press
Full text available: [pdf\(1.38 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Processors in large-scale multiprocessors must be able to tolerate large communication latencies and synchronization delays. This paper describes the architecture of a rapid-context-switching processor called APRIL with support for fine-grain threads and synchronization. APRIL achieves high single-thread performance and supports virtual dynamic threads. A commercial RISC-based implementation of APRIL and a run-time software system that can switch contexts in about 10 cycles is described. Me ...

91 **Problem and machine sensitive communication optimization**
 Thomas Fahringer, Eduard Mehofer
July 1998 **Proceedings of the 12th international conference on Supercomputing ICS '98**
Publisher: ACM Press
Full text available: [pdf\(1.21 MB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)

92 **A region-based compilation technique for dynamic compilers**
 Toshio Suganuma, Toshiaki Yasue, Toshio Nakatani

Method inlining and data flow analysis are two major optimization components for effective program transformations, but they often suffer from the existence of rarely or never executed code contained in the target method. One major problem lies in the assumption that the compilation unit is partitioned at method boundaries. This article describes the design and implementation of a region-based compilation technique in our dynamic optimization framework, in which the compiled regions are selected ...

Keywords: JIT compiler, Region-based compilation, dynamic compilation, on-stack replacement, partial inlining

93 Sensor networks and performance analysis: Impact of virtual execution environments on processor energy consumption and hardware adaptation 

Shiwen Hu, Lizy K. John

June 2006 **Proceedings of the second international conference on Virtual execution environments VEE '06**

Publisher: ACM Press

Full text available:  pdf(306.86 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

During recent years, microprocessor energy consumption has been surging and efforts to reduce power and energy have received a lot of attention. At the same time, virtual execution environments (VEEs), such as Java virtual machines, have grown in popularity. Hence, it is important to evaluate the impact of virtual execution environments on microprocessor energy consumption. This paper characterizes the energy and power impact of two important components of VEEs, Just-in-time(JIT) optimization an ...

Keywords: energy efficiency, hardware adaptation, power dissipation

94 Automatic compiler techniques for thread coarsening for multithreaded architectures 

Gary M. Zoppetti, Gagan Agrawal, Lori Pollock, Jose Nelson Amaral, Xinan Tang, Guang Gao
May 2000 **Proceedings of the 14th international conference on Supercomputing ICS '00**

Publisher: ACM Press

Full text available:  pdf(1.09 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Multithreaded architectures are emerging as an important class of parallel machines. By allowing fast context switching between threads on the same processor, these systems hide communication and synchronization latencies and allow scalable parallelism for dynamic and irregular applications. Thread partitioning is the most important task in compiling high-level languages for multithreaded architectures. Non-preemptive multithreaded architectures, which can be built from off-the-shelf compon ...

95 Reducing energy consumption of multiprocessor SoC architectures by exploiting memory bank locality 

Mahmut Taylan Kandemir

April 2006 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 11 Issue 2

Publisher: ACM Press

Full text available:  pdf(1.05 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The next generation embedded architectures are expected to accommodate multiple processors on the same chip. While this makes interprocessor communication less costly as compared to traditional high-end parallel machines, it also makes off-chip requests very costly. In particular, frequent off-chip memory accesses do not only increase execution cycles but also increase overall power consumption. One way of alleviating this power problem is to divide the off-chip memory into multiple banks, each ...

Keywords: Banked memory systems, bank locality, compiler optimization, energy

96 An annotation language for optimizing software libraries

Samuel Z. Guyer, Calvin Lin
December 1999 **ACM SIGPLAN Notices , Proceedings of the 2nd conference on Domain-specific languages PLAN '99**, Volume 35 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.34 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper introduces an annotation language and a compiler that together can customize a library implementation for specific application needs. Our approach is distinguished by its ability to exploit high level, domain-specific information in the customization process. In particular, the annotations provide semantic information that enables our compiler to analyze and optimize library operations as if they were primitives of a domain-specific language. Thus, our approach yields many of the ...

97 OpenMP on networks of workstations

Honghui Lu, Y. Charlie Hu, Willy Zwaenepoel

November 1998 **Proceedings of the 1998 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '98**

Publisher: IEEE Computer Society

Full text available:  pdf(202.91 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

We describe an implementation of a sizable subset of OpenMP on networks of workstations (NOWs). By extending the availability of OpenMP to NOWs, we overcome one of its primary drawbacks compared to MPI, namely lack of portability to environments other than hardware shared memory machines. In order to support OpenMP execution on NOWs, our compiler targets a software distributed shared memory system (DSM) which provides multi-threaded execution and memory consistency. This paper presents two contri ...

98 Session 4: communications libraries: Run-time and compile-time support for adaptive irregular problems

Shamik D. Sharma, Ravi Ponnusamy, Bongki Moon, Yuan Shin Hwang, Raja Das, Joel Saltz
November 1994 **Proceedings of the 1994 ACM/IEEE conference on Supercomputing Supercomputing '94**

Publisher: ACM Press

Full text available:  pdf(1.08 MB) Additional Information: [full citation](#), [abstract](#), [references](#)

In adaptive irregular problems, data arrays are accessed via indirection arrays, and data access patterns change during computation. Parallelizing such problems on distributed memory machines requires support for dynamic data partitioning, efficient preprocessing and fast data migration. This paper describes CHAOS, a library of efficient runtime primitives that provides such support. To demonstrate the effectiveness of the runtime support, two adaptive irregular applications have been paralleliz ...

99 Process migration

Dejan S. Milojićić, Fred Douglis, Yves Paindaveine, Richard Wheeler, Songnian Zhou
September 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 3

Publisher: ACM Press

Full text available:  pdf(1.24 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Process migration is the act of transferring a process between two machines. It enables dynamic load distribution, fault resilience, eased system administration, and data access locality. Despite these goals and ongoing research efforts, migration has not achieved widespread use. With the increasing deployment of distributed systems in general, and distributed operating systems in particular, process migration is again receiving more attention in both research and product development. As hi ...

Keywords: distributed operating systems, distributed systems, load distribution, process migration

100 [Launching the new era](#)

 Kazuhiro Fuchi, Robert Kowalski, Koichi Furukawa, Kazunori Ueda, Ken Kahn, Takashi Chikayama, Evan Tick
March 1993 **Communications of the ACM**, Volume 36 Issue 3

Publisher: ACM Press

Full text available:  [pdf\(3.45 MB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#), [review](#)

Results 81 - 100 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

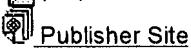
Terms used

[compiler optimization](#) [multi threading](#) [code motion](#) [critical section](#) [network processor](#)Found **60,664** of
199,787Sort results
by [Save results to a Binder](#)[Try an Advanced Search](#)Display
results [Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)Results 101 - 120 of 200 Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) **6** [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale **101 Automatic Thread Extraction with Decoupled Software Pipelining**

Guilherme Ottoni, Ram Rangan, Adam Stoler, David I. August

November 2005 **Proceedings of the 38th annual IEEE/ACM International Symposium
on Microarchitecture MICRO 38****Publisher:** IEEE Computer SocietyFull text available:  [pdf\(538.89 KB\)](#)Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Until recently, a steadily rising clock rate and other uniprocessor microarchitectural improvements could be relied upon to consistently deliver increasing performance for a wide range of applications. Current difficulties in maintaining this trend have lead microprocessor manufacturers to add value by incorporating multiple processors on a chip. Unfortunately, since decades of compiler research have not succeeded in delivering automatic threading for prevalent code properties, this approach dem ...

102 Scientific Computations on Modern Parallel Vector Systems

Leonid Oliker, Andrew Canning, Jonathan Carter, John Shalf, Stephane Ethier

November 2004 **Proceedings of the 2004 ACM/IEEE conference on Supercomputing SC
'04****Publisher:** IEEE Computer SocietyFull text available:  [pdf\(239.19 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#)

Computational scientists have seen a frustrating trend of stagnating application performance despite dramatic increases in the claimed peak capability of high performance computing systems. This trend has been widely attributed to the use of superscalar-based commodity components who's architectural designs offer a balance between memory performance, network capability, and execution rate that is poorly matched to the requirements of large-scale numerical computations. Recently, two innovative p ...

103 A pipelined memory architecture for high throughput network processors Timothy Sherwood, George Varghese, Brad CalderMay 2003 **ACM SIGARCH Computer Architecture News , Proceedings of the 30th
annual international symposium on Computer architecture ISCA '03, Volume
31 Issue 2****Publisher:** ACM PressFull text available:  [pdf\(213.66 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Designing ASICs for each new generation of backbone routers is a time intensive and fiscally draining process. In this paper we focus on the design of a programmable architecture for backbone routers, based on the manipulation of wide irregular memory words, that can provide a feasible design alternative to custom ASICs. We propose a pipelined memory design that emphasizes worst-case throughput over latency, and co-explore architectural tradeoffs with the design of several important network algo ...

104 Software implementation strategies for power-conscious systems

Kshirasagar Naik, David S. L. Wei

June 2001 **Mobile Networks and Applications**, Volume 6 Issue 3

Publisher: Kluwer Academic Publishers

Full text available:  pdf(238.01 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

A variety of systems with possibly embedded computing power, such as small portable robots, hand-held computers, and automated vehicles, have power supply constraints. Their batteries generally last only for a few hours before being replaced or recharged. It is important that all design efforts are made to conserve power in those systems. Energy consumption in a system can be reduced using a number of techniques, such as low-power electronics, architecture-level power reduction, compiler te ...

Keywords: energy saving, low power system, software design and implementation

105 From flop to megaflops: Java for technical computing

 José E. Moreira, Samuel P. Midkiff, Manish Gupta

March 2000 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 22 Issue 2

Publisher: ACM Press

Full text available:  pdf(371.84 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Although there has been some experimentation with Java as a language for numerically intensive computing, there is a perception by many that the language is unsuited for such work because of performance deficiencies. In this article we show how optimizing array bounds checks and null pointer checks creates loop nests on which aggressive optimizations can be used. Applying these optimizations by hand to a simple matrix-multiply test case leads to Java-compliant programs whose performance is ...

Keywords: arrays, compilers, java

106 Fortran 90D/HPF compiler for distributed memory MIMD computers: design,

 [implementation, and performance results](#)

Z. Bozkus, A. Choudhary, G. Fox, T. Haupt, S. Ranka

December 1993 **Proceedings of the 1993 ACM/IEEE conference on Supercomputing Supercomputing '93**

Publisher: ACM Press

Full text available:  pdf(971.18 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

107 Analysis and Performance Results of a Molecular Modeling Application on Merrimac

Mattan Erez, Jung Ho Ahn, Ankit Garg, William J. Dally, Eric Darve

November 2004 **Proceedings of the 2004 ACM/IEEE conference on Supercomputing SC '04**

Publisher: IEEE Computer Society

Full text available:  pdf(8.74 MB) Additional Information: [full citation](#), [abstract](#)

The Merrimac supercomputer uses stream processors and a high-radix network to achieve high performance at low cost and low power. The stream architecture matches the capabilities of modern semiconductor technology with compute-intensive parallel applications. We present a detailed case study of porting the GROMACS molecular-dynamics force calculation to Merrimac. The characteristics of the architecture which stress locality, parallelism, and decoupling of memory operations and computation, allow ...

108 A fast Fourier transform compiler

 Matteo Frigo

May 1999 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1999 conference on Programming language design and implementation PLDI '99**, Volume 34

Issue 5

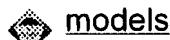
Publisher: ACM Press

Full text available:  pdf(1.48 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The FFTW library for computing the discrete Fourier transform (DFT) has gained a wide acceptance in both academia and industry, because it provides excellent performance on a variety of machines (even competitive with or faster than equivalent libraries supplied by vendors). In FFTW, most of the performance-critical code was generated automatically by a special-purpose compiler, called genfft, that outputs C code. Written in Objective Caml, genfft can produce DFT programs for any input length, a ...

109 Terascale spectral element dynamical core for atmospheric general circulation



Richard D. Loft, Stephen J. Thomas, John M. Dennis

November 2001 **Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '01**

Publisher: ACM Press

Full text available:  pdf(478.09 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Climate modeling is a grand challenge problem where scientific progress is measured not in terms of the largest problem that can be solved but by the highest achievable integration rate. These models have been notably absent in previous Gordon Bell competitions due to their inability to scale to large processor counts. A scalable and efficient spectral element atmospheric model is presented. A new semi-implicit time stepping scheme accelerates the integration rate relative to an explicit model b ...

110 Orchestrating shots for the national ignition facility



David G. Mathisen, Robert W. Carey

November 2005 **ACM SIGAda Ada Letters , Proceedings of the 2005 annual ACM SIGAda international conference on Ada: The Engineering of Correct and Reliable Software for Real-Time & Distributed Systems using Ada and Related Technologies SigAda '05**, Volume XXV Issue 4

Publisher: ACM Press

Full text available:  pdf(1.19 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The National Ignition Facility (NIF), currently under construction at the Lawrence Livermore National Laboratory, is a stadium-sized facility containing a 192-beam, 1.8 Megajoule, 500-Terawatt, ultra-violet laser system together with a 10-meter diameter target chamber with room for nearly 100 experimental diagnostics. When completed, NIF will be the world's largest and most energetic laser experimental system, providing an international center to study inertial confinement fusion and physics of ...

Keywords: Ada95, CORBA, XML, architecture, concurrency, data driven, framework, java, model-based, multi-threaded, state machine, workflow

111 ParADE: An OpenMP Programming Environment for SMP Cluster Systems

Yang-Suk Kee, Jin-Soo Kim, Soonhoi Ha

November 2003 **Proceedings of the 2003 ACM/IEEE conference on Supercomputing SC '03**

Publisher: IEEE Computer Society

Full text available:  pdf(244.73 KB) Additional Information: [full citation](#), [abstract](#), [citations](#)

Demand for programming environments to exploit clusters of symmetric multiprocessors (SMPs) is increasing. In this paper, we present a new programming environment, called ParADE, to enable easy, portable, and high-performance programming on SMP clusters. It is an OpenMP programming environment on top of a multi-threaded software distributed shared memory (SDSM) system with a variant of home-based lazy release consistency protocol. To boost performance, the runtime system provides explicit messag ...

Keywords: programming environment, SMP cluster, software distributedshared memory, hybrid programming, OpenMP, MPI

112 Special session on reconfigurable computing: The happy marriage of architecture and application in next-generation reconfigurable systems

Ingrid Verbauwhede, Patrick Schaumont

April 2004 **Proceedings of the 1st conference on Computing frontiers CF '04**

Publisher: ACM Press

Full text available:  pdf(398.28 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

New applications and standards are first conceived only for functional correctness and without concerns for the target architecture. The next challenge is to map them onto an architecture. Embedding such applications in a portable, low-energy context is the art of molding it onto an energy-efficient target architecture combined with an energy efficient execution. With a reconfigurable architecture, this task becomes a two-way process where the architecture adapts to the application and vice-vers ...

Keywords: embedded, real-time systems

113 System & architectural-level power optimization: Selective code/data migration for reducing communication energy in embedded MpSoC architectures

O. Ozturk, M. Kandemir, S. W. Son, M. Karakoy

April 2006 **Proceedings of the 16th ACM Great Lakes symposium on VLSI GLSVLSI '06**

Publisher: ACM Press

Full text available:  pdf(257.84 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Proliferation of embedded on-chip multiprocessor architectures (MpSoC) motivates researchers from both academia and industry to consider optimization techniques for such architectures. While many proposals on code/data partitioning on parallel architectures try to minimize interprocessor communication requirements at runtime, many applications still have significant runtime communication requirements. This paper proposes a novel task/data migration scheme that decides whether to migrate task or ...

Keywords: MPSoC, energy, migration

114 Computation techniques for FPGAs: An FPGA-based VLIW processor with custom hardware execution

Alex K. Jones, Raymond Hoare, Dara Kusic, Joshua Fazekas, John Foster

February 2005 **Proceedings of the 2005 ACM/SIGDA 13th international symposium on Field-programmable gate arrays FPGA '05**

Publisher: ACM Press

Full text available:  pdf(220.52 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The capability and heterogeneity of new FPGA (Field Programmable Gate Array) devices continues to increase with each new line of devices. Efficiently programming these devices is increasing in difficulty. However, FPGAs continue to be utilized for algorithms traditionally targeted to embedded DSP microprocessors such as signal and image processing applications. This paper presents an architecture that combines VLIW (Very Large Instruction Word) processing with the capability to introduce applicat ...

Keywords: NIOS, VLIW, compiler, kernels, parallelism, synthesis

115 Architecture: Supporting concurrent applications in wireless sensor networks

Yang Yu, Loren J. Rittle, Vartika Bhandari, Jason B. LeBrun

October 2006 **Proceedings of the 4th international conference on Embedded networked sensor systems SenSys '06**

Publisher: ACM Press

Full text available:  pdf(1.65 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

It is vital to support concurrent applications sharing a wireless sensor network in order to reduce the deployment and administrative costs, thus increasing the usability and efficiency of the network. We describe Melete¹, a system that supports concurrent applications with efficiency, reliability, flexibility, programmability, and scalability. Our

work is based on the Maté virtual machine [1] with significant modifications and enhancements. Melete enables reliable storage and ...

Keywords: Maté, concurrent applications, dynamic group formation, group-keyed code dissemination, group-keyed code distribution, melete, muse, network protocols, trickle, virtual machine, wireless sensor networks

116 Java bytecode to native code translation: the caffeine prototype and preliminary results

Cheng-Hsueh A. Hsieh, John C. Gyllenhaal, Wen-mei W. Hwu

December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture MICRO 29**

Publisher: IEEE Computer Society

Full text available:  pdf(1.03 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The Java bytecode language is emerging as a software distribution standard. With major vendors committed to porting the Java run-time environment to their platforms, programs in Java bytecode are expected to run without modification on multiple platforms. These first generation run-time environments rely on an interpreter to bridge the gap between the bytecode instructions and the native hardware. This interpreter approach is sufficient for specialized applications such as Internet browsers wher ...

117 Session 6: threads: TAPE: a transactional application profiling environment

 Hassan Chafi, Chi Cao Minh, Austen McDonald, Brian D. Carlstrom, JaeWoong Chung, Lance Hammond, Christos Kozyrakis, Kunle Olukotun

June 2005 **Proceedings of the 19th annual international conference on Supercomputing ICS '05**

Publisher: ACM Press

Full text available:  pdf(714.71 KB)

Additional Information: [full citation](#), [abstract](#), [references](#)

Transactional Coherence and Consistency (TCC) provides a new parallel programming model that uses transactions as the basic unit of parallel work and communication. TCC simplifies the development of correct parallel code because hardware provides transaction atomicity and ordering. Nevertheless, the programmer or a dynamic compiler must still optimize the parallel code for performance. This paper presents TAPE, a hardware and software infrastructure for profiling in TCC systems. TAPE extends the ...

118 Technical papers: Memory--Sequoia: programming the memory hierarchy

 Kayvon Fatahalian, Daniel Reiter Horn, Timothy J. Knight, Larkhoon Leem, Mike Houston, Ji Young Park, Mattan Erez, Manman Ren, Alex Aiken, William J. Dally, Pat Hanrahan

November 2006 **Proceedings of the 2006 ACM/IEEE conference on Supercomputing SC '06**

Publisher: ACM Press

Full text available:  pdf(231.75 KB)

Additional Information: [full citation](#), [abstract](#), [references](#)

 html(2.03 KB)

We present Sequoia, a programming language designed to facilitate the development of memory hierarchy aware parallel programs that remain portable across modern machines featuring different memory hierarchy configurations. Sequoia abstractly exposes hierarchical memory in the programming model and provides language mechanisms to describe communication vertically through the machine and to localize computation to particular memory locations within it. We have implemented a complete programming sy ...

119 Pinot: Speculative Multi-threading Processor Architecture Exploiting Parallelism over a Wide Range of Granularities

Taku Ohsawa, Masamichi Takagi, Shoji Kawahara, Satoshi Matsushita

November 2005 **Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture MICRO 38**

Publisher: IEEE Computer Society

Full text available:  pdf(592.97 KB)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

We propose a speculative multi-threading processor architecture called Pinot. Pinot exploits parallelism over a wide range of granularities without modifying program sources. Since exploitation of fine-grain parallelism suffers from limits of parallelism and overhead incurred by parallelization, it is better to extract coarse-grain parallelism. Coarse-grain parallelism is biased in some programs (mainly, numerical ones) and some program portions. Therefore, exploiting both coarse- and fine-grain ...

120 [Compilation, performance, and energy: Compilation for explicitly managed memory hierarchies](#) 

 Timothy J. Knight, Ji Young Park, Manman Ren, Mike Houston, Mattan Erez, Kayvon Fatahalian, Alex Aiken, William J. Dally, Pat Hanrahan

March 2007 **Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '07**

Publisher: ACM Press

Full text available:  pdf(495.68 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We present a compiler for machines with an explicitly managed memory hierarchy and suggest that a primary role of any compiler for such architectures is to manipulate and schedule a hierarchy of bulk operations at varying scales of the application and of the machine. We evaluate the performance of our compiler using several benchmarks running on a Cell processor.

Keywords: bulk operations, software-managed memory hierarchy

Results 101 - 120 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) **6** [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



Terms used

[compiler optimization multi threading code motion critical section network processor](#)Found **60,664** of
199,787

Sort results by

 [Save results to a Binder](#)Try an [Advanced Search](#)
Try this search in [The ACM Guide](#)

Display results

 [Search Tips](#)
 Open results in a new windowResults 121 - 140 of 200 Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale

121 [Power awareness: Combining compiler and runtime IPC predictions to reduce energy in next generation architectures](#)
 Saurabh Chheda, Osman Unsal, Israel Koren, C. Mani Krishna, Csaba Andras Moritz
April 2004 **Proceedings of the 1st conference on Computing frontiers CF '04**
Publisher: ACM PressFull text available: [pdf\(336.02 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Next generation architectures will require innovative solutions to reduce energy consumption. One of the trends we expect is more extensive utilization of compiler information directly targeting energy optimizations. As we show in this paper, static information provides some unique benefits, not available with runtime hardware-based techniques alone. To achieve energy reduction, we use IPC information at various granularities, to adaptively adjust voltage and speed, and to throttle the fetch rat ...

Keywords: adaptive voltage scaling, compiler architecture interaction, fetch throttling, instruction level parallelism, low power design

122 [A structural view of the Cedar programming environment](#)
 Daniel C. Swinehart, Polle T. Zellweger, Richard J. Beach, Robert B. Hagmann
August 1986 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 8 Issue 4
Publisher: ACM PressFull text available: [pdf\(6.32 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents an overview of the Cedar programming environment, focusing on its overall structure—that is, the major components of Cedar and the way they are organized. Cedar supports the development of programs written in a single programming language, also called Cedar. Its primary purpose is to increase the productivity of programmers whose activities include experimental programming and the development of prototype software systems for a high-performance personal computer. T ...

123 [Generation and quantitative evaluation of dataflow clusters](#)
 Lucas Roh, Walid A. Najjar, A. P. Wim Böhm
July 1993 **Proceedings of the conference on Functional programming languages and computer architecture FPCA '93**
Publisher: ACM PressFull text available: [pdf\(993.02 KB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)**124**

[Improved multithreading techniques for hiding communication latency in multiprocessors](#)

Publisher: ACM Press

Full text available:  pdf(1.13 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Shared memory multiprocessors are considered among the easiest parallel computers to program. However building shared memory machines with thousands of processors has proved difficult because of the inevitably long memory latencies. Much previous research has focused on cache coherency techniques, but it remains unclear if caches can obtain sufficiently high hit rates. In this paper we present improved multithreading techniques that can easily tolerate latencies of hundreds of cycles, and y ...

125 Performance comparison of ILP machines with cycle time evaluation 

 Tetsuya Hara, Hideki Ando, Chikako Nakanishi, Masao Nakaya

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture ISCA '96**, Volume 24 Issue 2

Publisher: ACM Press

Full text available:  pdf(1.48 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Many studies have investigated performance improvement through exploiting instruction-level parallelism (ILP) with a particular architecture. Unfortunately, these studies indicate performance improvement using the number of cycles that are required to execute a program, but do not quantitatively estimate the penalty imposed on the cycle time from the architecture. Since the performance of a microprocessor must be measured by its execution time, a cycle time evaluation is required as well as a cy ...

126 Mobile and distributed systems: Enabling Java mobile computing on the IBM Jikes 

 research virtual machine

Giacomo Cabri, Letizia Leonardi, Raffaele Quitadamo

August 2006 **Proceedings of the 4th international symposium on Principles and practice of programming in Java PPPJ '06**

Publisher: ACM Press

Full text available:  pdf(389.80 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Today's complex applications must face the distribution of data and code among different network nodes. Java is a wide-spread language that allows developers to build complex software, even distributed, but it cannot handle the migration of computations (i.e. threads), due to intrinsic limitations of many traditional JVMs. After analyzing the approaches in literature, this paper presents our research work on the IBM Jikes Research Virtual Machine: exploiting some of its innovative VM techniques, ...

Keywords: Java virtual machine, code mobility, distributed applications, thread persistence

127 Compile-time dynamic voltage scaling settings: opportunities and limits 

 Fen Xie, Margaret Martonosi, Sharad Malik

May 2003 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation PLDI '03**, Volume 38 Issue 5

Publisher: ACM Press

Full text available:  pdf(291.26 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

With power-related concerns becoming dominant aspects of hardware and software design, significant research effort has been devoted towards system power minimization. Among run-time power-management techniques, dynamic voltage scaling (DVS) has emerged as an important approach, with the ability to provide significant power savings. DVS exploits the ability to control the power consumption by varying a processor's supply voltage (V) and clock frequency (f). DVS controls energy by scheduling diffe ...

Keywords: analytical model, compiler, dynamic voltage scaling, low power, mixed-integer linear programming

128 MGS: a multigrain shared memory system

Donald Yeung, John Kubiatowicz, Anant Agarwal

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture ISCA '96**, Volume 24 Issue 2

Publisher: ACM Press

Full text available:  pdf(1.37 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Parallel workstations, each comprising 10-100 processors, promise cost-effective general-purpose multiprocessing. This paper explores the coupling of such small- to medium-scale shared memory multiprocessors through software over a local area network to synthesize larger shared memory systems. We call these systems Distributed Scalable Shared-memory Multiprocessors (DSSMPs). This paper introduces the design of a shared memory system that uses multiple granularities of sharing, and presents an imp ...

129 Dynamic languages symposium chair's welcome: Runtime synthesis of high-

performance code from scripting languages

Christopher Mueller, Andrew Lumsdaine

October 2006 **Companion to the 21st ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications OOPSLA '06**

Publisher: ACM Press

Full text available:  pdf(2.34 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Scripting languages are ubiquitous in modern software engineering and are often used as the sole language for application development. However, some applications, specifically scientific and multimedia applications, often have small sections of code that require a higher level of performance than the host language can deliver. In many cases, the algorithm being optimized is simple and has a clear mapping to hardware resources. But, without introducing an intermediate language, developers general ...

Keywords: Python, SIMD, chemical fingerprint, machine code, meta-programming, synthetic programming

130 The berkeley software MPEG-1 video decoder

Ketan Mayer-Patel, Brian C. Smith, Lawrence A. Rowe

February 2005 **ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)**, Volume 1 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.55 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This article reprises the description of the Berkeley software-only MPEG-1 video decoder originally published in the proceedings of the 1st International ACM Conference on Multimedia in 1993. The software subsequently became widely used in a variety of research systems and commercial products. Its main impact was to provide a platform for experimenting with streaming compressed video and to expose the strengths and weaknesses of software-only video decoding using general purpose computing archit ...

Keywords: MPEG, Video compression

131 Finding effective optimization phase sequences

Prasad Kulkarni, Wankang Zhao, Hwashin Moon, Kyunghwan Cho, David Whalley, Jack Davidson, Mark Bailey, Yunheung Paek, Kyle Gallivan

June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7

Publisher: ACM Press

Full text available:  pdf(703.12 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index](#)

It has long been known that a single ordering of optimization phases will not produce the best code for every application. This phase ordering problem can be more severe when generating code for embedded systems due to the need to meet conflicting constraints on time, code size, and power consumption. Given that many embedded application developers are willing to spend time tuning an application, we believe a viable approach is to allow the developer to steer the process of optimizing a function ...

Keywords: genetic algorithms, interactive compilation, phase ordering

132 A region-based compilation technique for a Java just-in-time compiler

◆ Toshio Suganuma, Toshiaki Yasue, Toshio Nakatani

May 2003 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation PLDI '03**, Volume 38

Issue 5

Publisher: ACM Press

Full text available:  pdf(158.62 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Method inlining and data flow analysis are two major optimization components for effective program transformations, however they often suffer from the existence of rarely or never executed code contained in the target method. One major problem lies in the assumption that the compilation unit is partitioned at method boundaries. This paper describes the design and implementation of a region-based compilation technique in our dynamic compilation system, in which the compiled regions are selected a ...

Keywords: dynamic compilers, on-stack replacement, partial inlining, region-based compilation

133 Scientific computing on the Itanium™ processor

◆ Bruce Greer, John Harrison, Greg Henry, Wei Li, Peter Tang

November 2001 **Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '01**

Publisher: ACM Press

Full text available:  pdf(177.31 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The 64-bit Intel® Itanium™ architecture is designed for high-performance scientific and enterprise computing, and the Itanium processor is its first silicon implementation. Features such as extensive arithmetic support, predication, speculation, and explicit parallelism can be used to provide a sound infrastructure for supercomputing. A large number of high-performance computer companies are offering Itanium™-based systems, some capable of peak performance exceeding 50 GFLOPS. In ...

Keywords: EPIC, fused multiply-add, itanium (TM) processor, linear algebra, transcendental functions

134 Multimedia and graphics: Enhancing loop buffering of media and telecommunications applications using low-overhead predication

John W. Sias, Hillary C. Hunter, Wen-mei W. Hwu

December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture MICRO 34**

Publisher: IEEE Computer Society

Full text available:  pdf(1.37 MB)  Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)
[Publisher Site](#)

Media- and telecommunications-focused processors, increasingly designed as deeply pipelined, statically-scheduled VLIWs, rely on loop buffers for low-overhead execution of simple loops. Key loops containing control flow pose a substantial problem---full predication has a high encoding overhead, and partial predication techniques do not support if-conversion, the transformation of general acyclic control flow into predicated blocks. Using a set of significant media processing benchmarks, drawn fr ...

135 Formalizing the safety of Java, the Java virtual machine, and Java card

◆ Pieter H. Hartel, Luc Moreau
December 2001 **ACM Computing Surveys (CSUR)**, Volume 33 Issue 4

Publisher: ACM Press

Full text available:  pdf(442.86 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We review the existing literature on Java safety, emphasizing formal approaches, and the impact of Java safety on small footprint devices such as smartcards. The conclusion is that although a lot of good work has been done, a more concerted effort is needed to build a coherent set of machine-readable formal models of the whole of Java and its implementation. This is a formidable task but we believe it is essential to build trust in Java safety, and thence to achieve ITSEC level 6 or Common Crite ...

Keywords: Common criteria, programming

136 A simple method for extracting models for protocol code

◆ David Lie, Andy Chou, Dawson Engler, David L. Dill
May 2001 **ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on Computer architecture ISCA '01**, Volume 29 Issue 2

Publisher: ACM Press

Full text available:  pdf(923.18 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The use of model checking for validation requires that models of the underlying system be created. Creating such models is both difficult and error prone and as a result, verification is rarely used despite its advantages. In this paper, we present a method for automatically extracting models from low level software implementations. Our method is based on the use of an extensible compiler system, xg++, to perform the extraction. The extracted model is combined with a model of the ha ...

137 Area-Performance Trade-offs in Tiled Dataflow Architectures

◆ Steven Swanson, Andrew Putnam, Martha Mercaldi, Ken Michelson, Andrew Petersen, Andrew Schwerin, Mark Oskin, Susan J. Eggers
May 2006 **ACM SIGARCH Computer Architecture News , Proceedings of the 33rd annual international symposium on Computer Architecture ISCA '06**, Volume 34 Issue 2

Publisher: IEEE Computer Society, ACM Press

Full text available:  pdf(487.22 KB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Tiled architectures, such as RAW, SmartMemories, TRIPS, and WaveScalar, promise to address several issues facing conventional processors, including complexity, wire-delay, and performance. The basic premise of these architectures is that larger, higher-performance implementations can be constructed by replicating the basic tile across the chip. This paper explores the area-performance trade-offs when designing one such tiled architecture, WaveScalar. We use a synthesizable RTL model and cycle-le ...

Keywords: WaveScalar, Dataflow computing, ASIC, RTL

138 An efficient implementation of Java's remote method invocation

◆ Jason Maassen, Rob van Nieuwpoort, Ronald Veldema, Henri E. Bal, Aske Plaat
May 1999 **ACM SIGPLAN Notices , Proceedings of the seventh ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '99**, Volume 34 Issue 8

Publisher: ACM Press

Full text available:  pdf(1.20 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Java offers interesting opportunities for parallel computing. In particular, Java Remote Method Invocation provides an unusually flexible kind of Remote Procedure Call. Unlike RPC, RMI supports polymorphism, which requires the system to be able to download

remote classes into a running application. Sun's RMI implementation achieves this kind of flexibility by passing around object type information and processing it at run time, which causes a major run time overhead. Using Sun's JDK 1.1.4 on a P ...

139 Synthesis and Design Tools: A compiler framework for mapping applications to a coarse-grained reconfigurable computer architecture

Girish Venkataramani, Walid Najjar, Fadi Kurdahi, Nader Bagherzadeh, Wim Bohm
November 2001 **Proceedings of the 2001 international conference on Compilers, architecture, and synthesis for embedded systems CASES '01**

Publisher: ACM Press

Full text available:  [pdf\(304.22 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The rapid growth of silicon densities has made it feasible to deploy reconfigurable hardware as a highly parallel computing platform. However, in most cases, the application needs to be programmed in hardware description or assembly languages, whereas most application programmers are familiar with the algorithmic programming paradigm. SA-C has been proposed as an expression-oriented language designed to implicitly express data parallel operations. Morphosys is a reconfigurable system-on-chip arc ...

140 The STAMPede approach to thread-level speculation

 J. Gregory Steffan, Christopher Colohan, Antonia Zhai, Todd C. Mowry
August 2005 **ACM Transactions on Computer Systems (TOCS)**, Volume 23 Issue 3

Publisher: ACM Press

Full text available:  [pdf\(1.72 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Multithreaded processor architectures are becoming increasingly commonplace: many current and upcoming designs support chip multiprocesssing, simultaneous multithreading, or both. While it is relatively straightforward to use these architectures to improve the throughput of a multithreaded or multiprogrammed workload, the real challenge is how to easily create *parallel software* to allow single programs to effectively exploit all of this raw performance potential. One promising technique fo ...

Keywords: Thread-level speculation, automatic parallelization, cache coherence, chip-multiprocessing

Results 121 - 140 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

THE ACM DIGITAL LIBRARY

 Terms used thread partitioning based on cost model

Found 127,417 of 199,915

 Sort results
by

 relevance 
 [Save results to a Binder](#)
[Try an Advanced Search](#)

 Display
results

 expanded form 
 [Search Tips](#)
[Try this search in The ACM Guide](#)
 [Open results in a new window](#)

Results 1 - 20 of 200

 Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale 
1 Thread partitioning and scheduling based on cost model
 Xinan Tang, J. Wang, Kevin B. Theobald, Guang R. Gao

 June 1997 **Proceedings of the ninth annual ACM symposium on Parallel algorithms and architectures SPAA '97**

Publisher: ACM Press

 Full text available:  [pdf\(1.49 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

2 How "hard" is thread partitioning and how "bad" is a list scheduling based partitioning algorithm?
 Xinan Tang, Guang R. Gao

 June 1998 **Proceedings of the tenth annual ACM symposium on Parallel algorithms and architectures SPAA '98**

Publisher: ACM Press

 Full text available:  [pdf\(1.27 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

3 Thread partitioning method for hardware compiler batch
 Mizuki Takahashi, Nagisa Ishiura, Akihisa Yamada, Takashi Kambe

 January 2000 **Proceedings of the 2000 conference on Asia South Pacific design automation ASP-DAC '00**

Publisher: ACM Press

 Full text available:  [pdf\(72.39 KB\)](#) Additional Information: [full citation](#), [references](#)

4 Case studies in embedded system design: A detailed cost model for concurrent use with hardware/software co-design
 Daniel Ragan, Peter Sandborn, Paul Stoaks

 June 2002 **Proceedings of the 39th conference on Design automation DAC '02**

Publisher: ACM Press

 Full text available:  [pdf\(278.42 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


Hardware/software co-design methodologies generally focus on the prediction of system performance or co-verification of system functionality. This study extends this conventional focus through the development of a methodology and software tool that evaluates system (hardware and software) development, fabrication, and testing costs (dollar costs) concurrent with hardware/software partitioning in a co-design environment. Based on the determination of key metrics such as gate count and lines of sof ...

Keywords: cost modeling, cost-performance trade-off

5 Compiling nested data-parallel programs for shared-memory multiprocessors

 Siddhartha Chatterjee

July 1993 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 15 Issue 3

Publisher: ACM Press

Full text available:  pdf(4.17 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#),
[review](#)

Keywords: compilers, data parallelism, shared-memory multiprocessors

6 Models and languages for parallel computation

 David B. Skillicorn, Domenico Talia

June 1998 **ACM Computing Surveys (CSUR)**, Volume 30 Issue 2

Publisher: ACM Press

Full text available:  pdf(298.05 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We survey parallel programming models and languages using six criteria to assess their suitability for realistic portable parallel programming. We argue that an ideal model should be easy to program, should have a software development methodology, should be architecture-independent, should be easy to understand, should guarantee performance, and should provide accurate information about the cost of programs. These criteria reflect our belief that developments in parallelism must be driven by ...

Keywords: general-purpose parallel computation, logic programming languages, object-oriented languages, parallel programming languages, parallel programming models, software development methods, taxonomy

7 Post-pass binary adaptation for software-based speculative precomputation

 Steve S.W. Liao, Perry H. Wang, Hong Wang, Gerolf Hoflehner, Daniel Lavery, John P. Shen
May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation PLDI '02**, Volume 37

Issue 5

Publisher: ACM Press

Full text available:  pdf(330.60 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Recently, a number of thread-based prefetching techniques have been proposed. These techniques aim at improving the latency of single-threaded applications by leveraging multithreading resources to perform memory prefetching via speculative prefetch threads. Software-based speculative precomputation (SSP) is one such technique, proposed for multithreaded Itanium models. SSP does not require expensive hardware support-instead it relies on the compiler to adapt binaries to perform prefetching on o ...

Keywords: chaining speculative precomputation, delay minimization, dependence reduction, long-range thread-based prefetching, loop rotation, pointer, post-pass, prediction, scheduling, slack, slicing, speculation, triggering

8 Automatic compiler techniques for thread coarsening for multithreaded architectures

 Gary M. Zoppetti, Gagan Agrawal, Lori Pollock, Jose Nelson Amaral, Xinan Tang, Guang Gao
May 2000 **Proceedings of the 14th international conference on Supercomputing ICS '00**

Publisher: ACM Press

Full text available:  pdf(1.09 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Multithreaded architectures are emerging as an important class of parallel machines. By allowing fast context switching between threads on the same processor, these systems hide communication and synchronization latencies and allow scalable parallelism for

dynamic and irregular applications. Thread partitioning is the most important task in compiling high-level languages for multithreaded architectures. Non-preemptive multithreaded architectures, which can be built from off-the-shelf compon ...

9 Chores: enhanced run-time support for shared-memory parallel computing

 Derek L. Eager, John Jahorjan
February 1993 **ACM Transactions on Computer Systems (TOCS)**, Volume 11 Issue 1

Publisher: ACM Press

Full text available:  pdf(2.24 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Parallel computing is increasingly important in the solution of large-scale numerical problems. The difficulty of efficiently hand-coding parallelism, and the limitations of parallelizing compilers, have nonetheless restricted its use by scientific programmers. In this paper we propose a new paradigm, chores, for the run-time support of parallel computing on shared-memory multiprocessors. We consider specifically uniform memory access shared-memory environments, ...

10 Helper threads via virtual multithreading on an experimental itanium® 2 processor-based platform

 Perry H. Wang, Jamison D. Collins, Hong Wang, Dongkeun Kim, Bill Greene, Kai-Ming Chan, Aamir B. Yunus, Terry Sych, Stephen F. Moore, John P. Shen
October 2004 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the 11th international conference on Architectural support for programming languages and operating systems ASPLOS-XI**, Volume 39 , 38 , 32 Issue 11 , 5 , 5

Publisher: ACM Press

Full text available:  pdf(225.47 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Helper threading is a technology to accelerate a program by exploiting a processor's multithreading capability to run ``assist'' threads. Previous experiments on hyper-threaded processors have demonstrated significant speedups by using helper threads to prefetch hard-to-predict delinquent data accesses. In order to apply this technique to processors that do not have built-in hardware support for multithreading, we introduce virtual multithreading (VMT), a novel form of switch-on-event user-level ...

Keywords: DB2 database, PAL, cache miss prefetching, helper thread, itanium processor, multithreading, switch-on-event

11 UML-based multiprocessor SoC design framework

 Tero Kangas, Petri Kukkala, Heikki Orsila, Erno Salminen, Marko Hännikäinen, Timo D. Hääläinen, Jouni Riihimäki, Kimmo Kuusilinna
May 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 2

Publisher: ACM Press

Full text available:  pdf(3.37 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes a complete design flow for multiprocessor systems-on-chips (SoCs) covering the design phases from system-level modeling to FPGA prototyping. The design of complex heterogeneous systems is enabled by raising the abstraction level and providing several system-level design automation tools. The system is modeled in a UML design environment following a new UML profile that specifies the practices for orthogonal application and architecture modeling. The design flow tools are gov ...

Keywords: UML 2.0, architecture exploration, design flow

12 Modeling methodology a: simulation methodologies: A non-fragmenting partitioning algorithm for hierarchical models

Roland Ewald, Jan Himmelsbach, Adelinde M. Uhrmacher
December 2006 **Proceedings of the 37th conference on Winter simulation WSC '06**

The simulation system JAMES II is aimed at supporting a range of modeling formalisms and simulation engines. The partitioning of models is essential for distributed simulation. A suitable partition depends on model, hardware, and simulation algorithm characteristics. Therefore, a partitioning layer has been created in JAMES II which allows to plug in partitioning algorithms on demand. Three different partitioning algorithms have been implemented. In addition to the well known Kernighan-Lin algor ...

13 An object-based programming model for shared data 

 Gail E. Kaiser, Brent Hailpern

April 1992 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 14 Issue 2

Publisher: ACM Press

Full text available:  pdf(3.28 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The classical object model supports private data within objects and clean interfaces between objects, and by definition does not permit sharing of data among arbitrary objects. This is a problem for real-world applications, such as advanced financial services and integrated network management, where the same data logically belong to multiple objects and may be distributed over multiple nodes on the network. Rather than give up the advantages of encapsulated objects in modeling real-world en ...

Keywords: coordination language, daemons, financial applications, object-based, real-time, sharing

14 Processor-pool-based scheduling for large-scale NUMA multiprocessors 

 Songnian Zhou, Timothy Brecht

April 1991 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1991 ACM SIGMETRICS conference on Measurement and modeling of computer systems SIGMETRICS '91**, Volume 19 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.26 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Large-scale Non-Uniform Memory Access (NUMA) multiprocessors are gaining increased attention due to their potential for achieving high performance through the replication of relatively simple components. Because of the complexity of such systems, scheduling algorithms for parallel applications are crucial in realizing the performance potential of these systems. In particular, scheduling methods must consider the scale of the system, with the increased likelihood of creating bottlenecks, along wi ...

15 Optimizing data aggregation for cluster-based internet services 

 Lingkun Chu, Hong Tang, Tao Yang, Kai Shen

June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '03**,

Volume 38 Issue 10

Publisher: ACM Press

Full text available:  pdf(275.38 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#), [review](#)

Large-scale cluster-based Internet services often host partitioned datasets to provide incremental scalability. The aggregation of results produced from multiple partitions is a fundamental building block for the delivery of these services. This paper presents the design and implementation of a programming primitive -- Data Aggregation Call (DAC) -- to exploit partition parallelism for cluster-based Internet services. A DAC request specifies a local processing operator and a global reduction ope ...

Keywords: cluster-based network services, fault tolerance, load-adaptive tree formation, response time, scalable data aggregation, throughput

Secure program partitioning

Steve Zdancewic, Lantian Zheng, Nathaniel Nystrom, Andrew C. Myers
August 2002 **ACM Transactions on Computer Systems (TOCS)**, Volume 20 Issue 3

Publisher: ACM Press

Full text available:  pdf(497.12 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents secure program partitioning, a language-based technique for protecting confidential data during computation in distributed systems containing mutually untrusted hosts. Confidentiality and integrity policies can be expressed by annotating programs with security types that constrain information flow; these programs can then be partitioned automatically to run securely on heterogeneously trusted hosts. The resulting communicating subprograms collectively implement the original p ...

Keywords: Confidentiality, declassification, distributed systems, downgrading, integrity, mutual distrust, secrecy, security policies, type systems

17 Mitosis compiler: an infrastructure for speculative threading based on pre-computation slices

 Carlos García Quiñones, Carlos Madriles, Jesús Sánchez, Pedro Marcuello, Antonio González, Dean M. Tullsen

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40

Issue 6

Publisher: ACM Press

Full text available:  pdf(487.97 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Speculative parallelization can provide significant sources of additional thread-level parallelism, especially for irregular applications that are hard to parallelize by conventional approaches. In this paper, we present the Mitosis compiler, which partitions applications into speculative threads, with special emphasis on applications for which conventional parallelizing approaches fail. The management of inter-thread data dependences is crucial for the performance of the system. The Mitosis fram ...

Keywords: automatic parallelization, pre-computation slices, speculative multithreading, thread-level parallelism

18 Learning-Based SMT Processor Resource Distribution via Hill-Climbing

 Seungryul Choi, Donald Yeung

May 2006 **ACM SIGARCH Computer Architecture News , Proceedings of the 33rd annual international symposium on Computer Architecture ISCA '06**, Volume 34 Issue 2

Publisher: IEEE Computer Society, ACM Press

Full text available:  pdf(823.43 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)

The key to high performance in Simultaneous Multithreaded (SMT) processors lies in optimizing the distribution of shared resources to active threads. Existing resource distribution techniques optimize performance only indirectly. They infer potential performance bottlenecks by observing indicators, like instruction occupancy or cache miss counts, and take actions to try to alleviate them. While the corrective actions are designed to improve performance, their actual performance impact is not kno ...

19 Minimum cost adaptive synchronization: experiments with the ParaSol system

 Edward Mascarenhas, Felipe Knop, Reuben Pasquini, Vernon Rego

October 1998 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 8 Issue 4

Publisher: ACM Press

Full text available:  pdf(265.07 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a novel adaptive synchronization algorithm, called the minimum average cost (MAC) algorithm, in the context of the parasol parallel simulation system. ParaSol is a

multithreaded system for parallel simulation on shared- and distributed-memory environments, designed to support domain-specific Simulation Object Libraries. The proposed MAC algorithm is based on minimizing the cost of synchronization delay and rollback at a process, whenever its simulation driver must decide whether ...

Keywords: ParaSol, adaptive synchronization, optimal delay, optimistic synchronization, parallel and distributed simulation, stochastic simulation, thread

20 Feature-Based Decomposition of Inductive Proofs Applied to Real-Time Avionics Software: An Experience Report

Vu Ha, Murali Rangarajan, Darren Cofer, Harald Rues, Bruno Dutertre

May 2004 **Proceedings of the 26th International Conference on Software Engineering ICSE '04**

Publisher: IEEE Computer Society

Full text available:  [pdf\(253.19 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

The hardware and software in modern aircraft controlsystems are good candidates for verification using formalmethods: they are complex, safety-critical, and challengethe capabilities of test-based verification strategies. Wehave previously reported on our use of model checking toverify the time partitioning property of the DeosŁ real-timeoperating system for embedded avionics. The size and complexityof this system have limited us to analyzing only oneconfiguration at a time. To overcome this lim ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)Terms used thread partitioning based on cost model

Found 127,417 of 199,915

Sort results by

 Save results to a Binder

Display results

 Search Tips Open results in a new window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 21 - 40 of 200

Result page: [previous](#)

1

2

3

4

5

6

7

8

9

10

[next](#)

Best 200 shown

Relevance scale

21 A cost-driven compilation framework for speculative parallelization of sequential programs Zhao-Hui Du, Chu-Cheow Lim, Xiao-Feng Li, Chen Yang, Qingyu Zhao, Tin-Fook Ngai
June 2004 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2004 conference on Programming language design and implementation PLDI '04**, Volume 39 Issue 6**Publisher:** ACM PressFull text available: [pdf\(235.14 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The emerging hardware support for thread-level speculation opens new opportunities to parallelize sequential programs beyond the traditional limits. By speculating that many data dependences are unlikely during runtime, consecutive iterations of a sequential loop can be executed speculatively in parallel. Runtime parallelism is obtained when the speculation is correct. To take full advantage of this new execution model, a program needs to be programmed or compiled in such a way that it exhibits ...

Keywords: cost-driven compilation, loop transformation, speculative multithreading, speculative parallel threading, speculative parallelization, thread-level speculation

22 Connectivity-based garbage collection Martin Hirzel, Amér Diwan, Matthew Hertz
October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '03**, Volume 38 Issue 11**Publisher:** ACM PressFull text available: [pdf\(521.65 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We introduce a new family of connectivity-based garbage collectors (Cbgc) that are based on potential object-connectivity properties. The key feature of these collectors is that the placement of objects into partitions is determined by performing one of several forms of connectivity analyses on the program. This enables partial garbage collections, as in generational collectors, but without the need for any write barrier. The contributions of this paper are 1) a novel family of garbage c ...

Keywords: connectivity based garbage collection

23 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research CASCON '97****Publisher:** IBM PressFull text available: [pdf\(4.21 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

24 Verification of time partitioning in the DEOS scheduler kernel

 John Penix, Willem Visser, Eric Engstrom, Aaron Larson, Nicholas Weininger
June 2000 **Proceedings of the 22nd international conference on Software engineering ICSE '00**

Publisher: ACM Press

Full text available:  pdf(111.58 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes an experiment to use the Spin model checking system to support automated verification of time partitioning in the Honeywell DEOS real-time scheduling kernel. The goal of the experiment was to investigate whether model checking could be used to find a subtle implementation error that was originally discovered and fixed during the standard formal review process. To conduct the experiment, a core slice of the DEOS scheduling kernel was first translated without abstraction ...

25 LoGPC: modeling network contention in message-passing programs

 Csaba Andras Moritz, Matthew I. Frank
June 1998 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems SIGMETRICS '98/PERFORMANCE '98**, Volume 26 Issue 1

Publisher: ACM Press

Full text available:  pdf(1.41 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In many real applications, for example those with frequent and irregular communication patterns or those using large messages, network contention and contention for message processing resources can be a significant part of the total execution time. This paper presents a new cost model, called LoGPC, that extends the LogP [9] and LogGP [4] models to account for the impact of network contention and network interface DMA behavior on the performance of message-passing programs. We validate LoGPC by a ...

26 A model for dataflow based vector execution

 W. Marcus Miller, Walid A. Najjar, A. P. Wim Böhm
July 1994 **Proceedings of the 8th international conference on Supercomputing ICS '94**

Publisher: ACM Press

Full text available:  pdf(1.28 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Although the dataflow model has been shown to allow the exploitation of parallelism at all levels, research of the past decade has revealed several fundamental problems: Synchronization at the instruction level, token matching, coloring and re-labeling operations have a negative impact on performance by significantly increasing the number of non-compute "overhead" cycles. Recently, many novel Hybrid von-Neumann Data Driven machines have been proposed to alleviate some of these p ...

27 Query evaluation techniques for large databases

 Goetz Graefe
June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2

Publisher: ACM Press

Full text available:  pdf(9.37 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Database management systems will continue to manage large data volumes. Thus, efficient algorithms for accessing and manipulating large sets and sequences will be required to provide acceptable performance. The advent of object-oriented and extensible database systems will not solve this problem. On the contrary, modern data models exacerbate the problem: In order to manipulate large sets of complex objects as

efficiently as today's database systems manipulate simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database systems, iterators, object-oriented database systems, operator model of parallelization, parallel algorithms, relational database systems, set-matching algorithms, sort-hash duality

28 Quick-VDR: Interactive View-Dependent Rendering of Massive Models

Sung-Eui Yoon, Brian Salomon, Russell Gayle, Dinesh Manocha

October 2004 **Proceedings of the conference on Visualization '04 VIS '04**

Publisher: IEEE Computer Society

Full text available:  pdf(379.11 KB) Additional Information: [full citation](#), [abstract](#), [citations](#)

We present a novel approach for interactive view-dependent rendering of massive models. Our algorithm combines view-dependent simplification, occlusion culling, and out-of-core rendering. We represent the model as a clustered hierarchy of progressive meshes (CHPM). We use the cluster hierarchy for coarse-grained selective refinement and progressive meshes for fine-grained local refinement. We present an out-of-core algorithm for computation of a CHPM that includes cluster decomposition, hierarch ...

Keywords: Interactive display, view-dependent rendering, occlusion culling, external-memory algorithm, levels-of-detail

29 Fine-grain parallelism with minimal hardware support: a compiler-controlled threaded abstract machine

David E. Culler, Anurag Sah, Klaus E. Schauser, Thorsten von Eicken, John Wawrzynek
April 1991 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating**

Systems Review , ACM SIGPLAN Notices , Proceedings of the fourth international conference on Architectural support for programming languages and operating systems ASPLOS-IV, Volume 19 , 25 , 26 Issue 2 , Special Issue , 4

Publisher: ACM Press

Full text available:  pdf(1.41 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

30 Automatically partitioning packet processing applications for pipelined architectures

Jinquan Dai, Bo Huang, Long Li, Luddy Harrison

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40

Issue 6

Publisher: ACM Press

Full text available:  pdf(541.83 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Modern network processors employs parallel processing engines (PEs) to keep up with explosive internet packet processing demands. Most network processors further allow processing engines to be organized in a pipelined fashion to enable higher processing throughput and flexibility. In this paper, we present a novel program transformation technique to exploit parallel and pipelined computing power of modern network processors. Our proposed method automatically partitions a sequential packet proces ...

Keywords: live-set transmission, network processor, packet processing, parallel, pipelining transformation, program partition

31 The costs and limits of availability for replicated services

Haifeng Yu, Amin Vahdat

February 2006 **ACM Transactions on Computer Systems (TOCS)**, Volume 24 Issue 1

Publisher: ACM Press

Full text available:  pdf(718.65 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

As raw system performance continues to improve at exponential rates, the utility of many services is increasingly limited by availability rather than performance. A key approach to improving availability involves replicating the service across multiple, wide-area sites. However, replication introduces well-known trade-offs between service consistency and availability. Thus, this article explores the benefits of dynamically trading consistency for availability using a *continuous consistency model* ...

Keywords: Availability, continuous consistency, network services, replication, trade-off, upper bound

32 Automating real-time multi-threaded application development

C. Riva, M. Krieger

November 1995 **Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research CASCON '95**

Publisher: IBM Press

Full text available:  pdf(89.57 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Over the past decade, the advances made in VLSI devices have driven the development of cost-effective, symmetric multiprocessor architectures. In turn, operating systems have evolved to take advantage of the true concurrency offered by these architectures. However, because of the lack of adequate development tools, delivery of the intrinsic benefits of these platforms directly to applications has proven elusive. This has been particularly the case for the use of these systems in real-time applic ...

33 A hybrid execution model for fine-grained languages on distributed memory

multicomputers

John Plevyak, Vijay Karamcheti, Xingbin Zhang, Andrew A. Chien

December 1995 **Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM) - Volume 00 Supercomputing '95**

Publisher: ACM Press

Full text available:  html(59.68 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)
 ps(419.70 KB)

While fine-grained concurrent languages can naturally capture concurrency in many irregular and dynamic problems, their flexibility has generally resulted in poor execution efficiency. In such languages the computation consists of many small threads which are created dynamically and synchronized implicitly. In order to minimize the overhead of these operations, we propose a hybrid execution model which dynamically adapts to runtime data layout, providing both sequential efficiency and low overhead ...

34 Supporting dynamic data structures on distributed-memory machines

Anne Rogers, Martin C. Carlisle, John H. Reppy, Laurie J. Hendren

March 1995 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 17 Issue 2

Publisher: ACM Press

Full text available:  pdf(2.05 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Compiling for distributed-memory machines has been a very active research area in recent years. Much of this work has concentrated on programs that use arrays as their primary data structures. To date, little work has been done to address the problem of supporting programs that use pointer-based dynamic data structures. The techniques developed for supporting SPMD execution of array-based programs rely on the fact that arrays are statically defined and directly addressable. Recursive data structures ...

Keywords: dynamic data structures

35 Automatic Thread Extraction with Decoupled Software Pipelining

Guilherme Ottoni, Ram Rangan, Adam Stoler, David I. August

November 2005 **Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture MICRO 38**

Publisher: IEEE Computer Society

Until recently, a steadily rising clock rate and other uniprocessor microarchitectural improvements could be relied upon to consistently deliver increasing performance for a wide range of applications. Current difficulties in maintaining this trend have lead microprocessor manufacturers to add value by incorporating multiple processors on a chip. Unfortunately, since decades of compiler research have not succeeded in delivering automatic threading for prevalent code properties, this approach dem ...

36 The state of the art in distributed query processing

 Donald Kossmann

December 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 4

Publisher: ACM Press

Full text available:  pdf(455.39 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Distributed data processing is becoming a reality. Businesses want to do it for many reasons, and they often must do it in order to stay competitive. While much of the infrastructure for distributed data processing is already there (e.g., modern network technology), a number of issues make distributed data processing still a complex undertaking: (1) distributed systems can become very large, involving thousands of heterogeneous sites including PCs and mainframe server machines; (2) the stat ...

Keywords: caching, client-server databases, database application systems, dissemination-based information systems, economic models for query processing, middleware, multitier architectures, query execution, query optimization, replication, wrappers

37 Converting thread-level parallelism to instruction-level parallelism via simultaneous multithreading

 Jack L. Lo, Joel S. Emer, Henry M. Levy, Rebecca L. Stamm, Dean M. Tullsen, S. J. Eggers August 1997 **ACM Transactions on Computer Systems (TOCS)**, Volume 15 Issue 3

Publisher: ACM Press

Full text available:  pdf(526.39 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

To achieve high performance, contemporary computer systems rely on two forms of parallelism: instruction-level parallelism (ILP) and thread-level parallelism (TLP). Wide-issue super-scalar processors exploit ILP by executing multiple instructions from a single program in a single cycle. Multiprocessors (MP) exploit TLP by executing different threads in parallel on different processors. Unfortunately, both parallel processing styles statically partition processor resources, thus preventing t ...

Keywords: cache interference, instruction-level parallelism, multiprocessors, multithreading, simultaneous multithreading, thread-level parallelism

38 Operating systems for sensor networks: Design and implementation of a single system image operating system for ad hoc networks

 Hongzhou Liu, Tom Roeder, Kevin Walsh, Rimon Barr, Emin Gün Sirer

June 2005 **Proceedings of the 3rd international conference on Mobile systems, applications, and services MobiSys '05**

Publisher: ACM Press

Full text available:  pdf(261.28 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

In this paper, we describe the design and implementation of a distributed operating system for ad hoc networks. Our system simplifies the programming of ad hoc networks and extends total system lifetime by making the entire network appear as a single virtual machine. It automatically and transparently partitions applications into components and dynamically finds them a placement on nodes within the network to reduce energy consumption and to increase system longevity. This paper describes our pr ...

39 Parallel execution of prolog programs: a survey

Gopal Gupta, Enrico Pontelli, Khayri A.M. Ali, Mats Carlsson, Manuel V. Hermenegildo
July 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 23 Issue 4

Publisher: ACM Press

Full text available: [pdf\(1.95 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Since the early days of logic programming, researchers in the field realized the potential for exploitation of parallelism present in the execution of logic programs. Their high-level nature, the presence of nondeterminism, and their referential transparency, among other characteristics, make logic programs interesting candidates for obtaining speedups through parallel execution. At the same time, the fact that the typical applications of logic programming frequently involve irregular computation ...

Keywords: Automatic parallelization, constraint programming, logic programming, parallelism, prolog

40 Facial modeling and animation

Jörg Haber, Demetri Terzopoulos
August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**

Publisher: ACM Press

Full text available: [pdf\(18.15 MB\)](#) Additional Information: [full citation](#), [abstract](#)

In this course we present an overview of the concepts and current techniques in facial modeling and animation. We introduce this research area by its history and applications. As a necessary prerequisite for facial modeling, data acquisition is discussed in detail. We describe basic concepts of facial animation and present different approaches including parametric models, performance-, physics-, and learning-based methods. State-of-the-art techniques such as muscle-based facial animation, mass-s ...

Results 21 - 40 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads: [Adobe Acrobat](#) [QuickTime](#) [Windows Media Player](#) [Real Player](#)

Terms used thread partitioning based on cost model

Found 127,417 of 199,915

Sort results
by
 relevance 
 [Save results to a Binder](#)[Try an Advanced Search](#)Display
results
 expanded form 
 [Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)

Results 41 - 60 of 200

Result page: [previous](#)

1

2

3

4

5

6

7

8

9

10

[next](#)

Best 200 shown

Relevance scale **41 A survey of processors with explicit multithreading** Theo Ungerer, Borut Robič, Jurij ŠilcMarch 2003 **ACM Computing Surveys (CSUR)**, Volume 35 Issue 1**Publisher:** ACM PressFull text available:  [pdf\(920.16 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Hardware multithreading is becoming a generally applied technique in the next generation of microprocessors. Several multithreaded processors are announced by industry or already into production in the areas of high-performance microprocessors, media, and network processors. A multithreaded processor is able to pursue two or more threads of control in parallel within the processor pipeline. The contexts of two or more threads of control are often stored in separate on-chip register sets. Unused i ...

Keywords: Blocked multithreading, interleaved multithreading, simultaneous multithreading

42 Cluster-based scalable network services Armando Fox, Steven D. Gribble, Yatin Chawathe, Eric A. Brewer, Paul GauthierOctober 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles SOSP '97**, Volume 31 Issue 5**Publisher:** ACM PressFull text available:  [pdf\(2.42 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)**43 Waiting algorithms for synchronization in large-scale multiprocessors** Beng-Hong Lim, Anant AgarwalAugust 1993 **ACM Transactions on Computer Systems (TOCS)**, Volume 11 Issue 3**Publisher:** ACM PressFull text available:  [pdf\(2.72 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Through analysis and experiments, this paper investigates two-phase waiting algorithms to minimize the cost of waiting for synchronization in large-scale multiprocessors. In a two-phase algorithm, a thread first waits by polling a synchronization variable. If the cost of polling reaches a limit Lpoll and further waiting is necessary, the thread is blocked, incurring an additional fixed cost, B. The choice of Lpoll

Keywords: barriers, blocking, competitive analysis, locks, producer-consumer synchronization, spinning, waiting time



services
Haifeng Yu, Amin Vahdat

August 2002 **ACM Transactions on Computer Systems (TOCS)**, Volume 20 Issue 3

Publisher: ACM Press

Full text available: [pdf\(406.85 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The tradeoffs between consistency, performance, and availability are well understood. Traditionally, however, designers of replicated systems have been forced to choose from either strong consistency guarantees or none at all. This paper explores the semantic space between traditional strong and optimistic consistency models for replicated services. We argue that an important class of applications can tolerate relaxed consistency, but benefit from bounding the maximum rate of inconsistent access ...

Keywords: Conit, consistency model, continuous consistency, network services, relaxed consistency, replication

45 [Modeling methodology b: Simulation and verification II: event-triggered environments for verification of real-time systems](#)

Darren D. Cofer, Murali Rangarajan

December 2003 **Proceedings of the 35th conference on Winter simulation: driving innovation WSC '03**

Publisher: Winter Simulation Conference

Full text available: [pdf\(395.81 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

The growing complexity and the safety-critical requirements of the embedded software in avionics systems present many challenges to current test-based verification technology. The use of formal verification methods can increase design assurance by exploring a larger range of system behaviors and fault conditions than can feasibly be covered by testing or simulation. However, one of the most challenging tasks faced in any formal verification activity is the construction of an adequate model fo ...

46 [Run-time adaptation in river](#)

Remzi H. Arpacı-Dusseau

February 2003 **ACM Transactions on Computer Systems (TOCS)**, Volume 21 Issue 1

Publisher: ACM Press

Full text available: [pdf\(849.04 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We present the design, implementation, and evaluation of run-time adaptation within the River dataflow programming environment. The goal of the River system is to provide adaptive mechanisms that allow database query-processing applications to cope with performance variations that are common in cluster platforms. We describe the system and its basic mechanisms, and carefully evaluate those mechanisms and their effectiveness. In our analysis, we answer four previously unanswered and important que ...

Keywords: Performance availability, clusters, parallel I/O, performance faults, robust performance, run-time adaptation

47 [Partitioning and Exploration Strategies in the TOSCA Co-Design Flow](#)

A. Balboni, W. Fornaciari, D. Sciuto

March 1996 **Proceedings of the 4th International Workshop on Hardware/Software Co-Design CODES '96**

Publisher: IEEE Computer Society

Full text available: [pdf\(900.37 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [citations](#)

[Publisher Site](#)

The TOSCA environment for hardware/software co-design of control dominated systems implemented on a single chip includes a novel approach to the system exploration phase for the evaluation of alternative architectures. The paper presents the metrics and the partitioning algorithm defined for the identification of the best hardware and software bindings and modularization, given the design constraints and goals. The system exploration phase is implemented as an iterative process directed by the u ...

48 [Exploiting perception in high-fidelity virtual environments: Exploiting perception in high-fidelity virtual environments](#)

 **Additional presentations from the 24th course are available on the citation page**

Mashhudha Glencross, Alan G. Chalmers, Ming C. Lin, Miguel A. Otaduy, Diego Gutierrez
July 2006 **ACM SIGGRAPH 2006 Courses SIGGRAPH '06**

Publisher: ACM Press

Full text available:  [pdf\(5.07 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)
 [mov\(68:6 MIN\)](#)

The objective of this course is to provide an introduction to the issues that must be considered when building high-fidelity 3D engaging shared virtual environments. The principles of human perception guide important development of algorithms and techniques in collaboration, graphical, auditory, and haptic rendering. We aim to show how human perception is exploited to achieve realism in high fidelity environments within the constraints of available finite computational resources. In this course w ...

Keywords: collaborative environments, haptics, high-fidelity rendering, human-computer interaction, multi-user, networked applications, perception, virtual reality

49 [Analytical cache models with applications to cache partitioning](#)

 G. Edward Suh, Srinivas Devadas, Larry Rudolph
June 2001 **Proceedings of the 15th international conference on Supercomputing ICS '01**

Publisher: ACM Press

Full text available:  [pdf\(854.26 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

An accurate, tractable, analytic cache model for time-shared systems is presented, which estimates the overall cache miss-rate of a multiproCESSing system with any cache size and time quanta. The input to the model consists of the isolated miss-rate curves for each process, the time quanta for each of the executing processes, and the total cache size. The output is the overall miss-rate. Trace-driven simulations demonstrate that the estimated miss-rate is very accurate. Since the model provid ...

50 [Modeling and evaluation of hardware/software designs](#)

 Neal K. Tibrewala, JoAnn M. Paul, Donald E. Thomas
April 2001 **Proceedings of the ninth international symposium on Hardware/software codesign CODES '01**

Publisher: ACM Press

Full text available:  [pdf\(703.44 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We introduce the foundation of a system modeling environment targeted at capturing the *anticipated interactions* of hardware and software behaviors — not just their co-execution. Key to our approach is the separation of external and internal design testbenches. We use a frequency interleaved scheduling foundation ideally suited to our approach because it allows unrestricted hardware and software modeling, a mix of untimed and timed software, and a layered approach using software s ...

Keywords: computer system modeling and simulation, digital system design, hardware/software codesign

51 [Decentralizing execution of composite web services](#)

 Mangala Gowri Nanda, Satish Chandra, Vivek Sarkar
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '04**, Volume 39 Issue 10

Publisher: ACM Press

Full text available:  [pdf\(429.43 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Distributed enterprise applications today are increasingly being built from services available over the web. A unit of functionality in this framework is a web service, a software application that exposes a set of "typed" connections that can be accessed over the web using standard protocols. These units can then be composed into a <i>composite</i> web service. BPEL (Business Process Execution Language) is a high-level distributed programming language for creating composite web servi ...

Keywords: business process, program dependence graph, work flow

52 Models for performance perturbation analysis

 Allen D. Malony, Daniel A. Reed

December 1991 **ACM SIGPLAN Notices , Proceedings of the 1991 ACM/ONR workshop on Parallel and distributed debugging PADD '91**, Volume 26 Issue 12

Publisher: ACM Press

Full text available:  pdf(1.16 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

53 System-level power optimization: techniques and tools

 Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Publisher: ACM Press

Full text available:  pdf(385.22 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic systems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, and storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survey ...

54 Modeling cost/performance of a parallel computer simulator

 Babak Falsafi, David A. Wood

January 1997 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 7 Issue 1

Publisher: ACM Press

Full text available:  pdf(585.68 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#), [review](#)

Keywords: conservative discrete-event simulation, memory systems, performance analysis, shared-memory multiprocessors

55 Formal Aspects and Distributed Systems: Modeling and simulation of steady state and transient behaviors for emergent SoCs

 JoAnn M. Paul, Arne J. Suppé, Donald E. Thomas

September 2001 **Proceedings of the 14th international symposium on Systems synthesis ISSS '01**

Publisher: ACM Press

Full text available:  pdf(409.23 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We introduce a formal basis for viewing computer systems as mixed steady state and non-steady state (transient) behaviors to motivate novel design strategies resulting from simultaneous consideration of function, scheduling and architecture. We relate three design styles: hierarchical decomposition, static mapping and directed platform that have traditionally been separate. By considering them together, we reason that once a steady state system is mapped to an architecture, the unused processing ...

Keywords: computer system modeling and simulation, hardware/software codesign,

56 A task- and data-parallel programming language based on shared objects

 Saniya Ben Hassen, Henri E. Bal, Ceriel J. H. Jacobs

November 1998 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 20 Issue 6

Publisher: ACM Press

Full text available:  pdf(434.44 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Many programming languages support either task parallelism, but few languages provide a uniform framework for writing applications that need both types of parallelism or data parallelism. We present a programming language and system that integrates task and data parallelism using shared objects. Shared objects may be stored on one processor or may be replicated. Objects may also be partitioned and distributed on several processors. Task parallelism is achieved by forking processes remotely a ...

Keywords: data parallelism, shared objects, task parallelism

57 Real-time shading

 Marc Olano, Kurt Akeley, John C. Hart, Wolfgang Heidrich, Michael McCool, Jason L. Mitchell, Randi Rost

August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**

Publisher: ACM Press

Full text available:  pdf(7.39 MB) Additional Information: [full citation](#), [abstract](#)

Real-time procedural shading was once seen as a distant dream. When the first version of this course was offered four years ago, real-time shading was possible, but only with one-of-a-kind hardware or by combining the effects of tens to hundreds of rendering passes. Today, almost every new computer comes with graphics hardware capable of interactively executing shaders of thousands to tens of thousands of instructions. This course has been redesigned to address today's real-time shading capabili ...

58 Tartan: evaluating spatial computation for whole program execution

 Mahim Mishra, Timothy J. Callahan, Tiberiu Chelcea, Girish Venkataramani, Seth C. Goldstein, Mihai Budiu

October 2006 **ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , ACM SIGPLAN Notices , Proceedings of the 12th international conference on Architectural support for programming languages and operating systems ASPLOS-XII**, Volume 40 , 34 , 41 Issue 5 , 5 , 11

Publisher: ACM Press

Full text available:  pdf(318.86 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Spatial Computing (SC) has been shown to be an energy-efficient model for implementing program kernels. In this paper we explore the feasibility of using SC for more than small kernels. To this end, we evaluate the performance and energy efficiency of entire applications on Tartan, a general-purpose architecture which integrates a reconfigurable fabric (RF) with a superscalar core. Our compiler automatically partitions and compiles an application into an instruction stream for the core and a con ...

Keywords: asynchronous circuits, dataflow machine, defect tolerance, low power, reconfigurable hardware, spatial computation

59 ESys.Net: a new solution for embedded systems modeling and simulation

 J. Lapalme, E. M. Aboulhamid, G. Nicolescu, L. Charest, F. R. Boyer, J. P. David, G. Bois June 2004 **ACM SIGPLAN Notices , Proceedings of the 2004 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems LCTES '04**, Volume 39 Issue 7

Publisher: ACM Press

Full text available: Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index](#)

The next generation of tools for embedded systems design will represent a common arena for several cooperating groups. These tools will permit system design at a high abstraction level and enable automatic refinement through several abstraction levels to obtain the final prototype. To facilitate this evolution, we propose a new .Net Framework based system level modeling and simulation environment. This environment allows (1) cooperation -- by enabling web-based design and multi-language features ...

Keywords: .Net, C#, C++, CIL, ESys.Net, HDLs, Java, SystemC, SystemVerilog, VHDL, Verilog, attribute programming, component-based programming, embedded systems, framework, hardware/software codesign, modeling, simulation, system on chip

60 Scheduling threads for low space requirement and good locality

 Girija J. Narlikar

June 1999 **Proceedings of the eleventh annual ACM symposium on Parallel algorithms and architectures SPAA '99**

Publisher: ACM Press

Full text available:  pdf(1.69 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

Results 41 - 60 of 200

Result page: [previous](#) [1](#) [2](#) **3** [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used thread partitioning based on cost model

Found 127,417 of 199,915

Sort results
by Save results to a BinderDisplay
results Search Tips Open results in a new
window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 61 - 80 of 200

Result page: previous

1

2

3

4

5

6

7

8

9

10

next

Best 200 shown

Relevance scale

61 Thread-based programming for the EM-4 hybrid dataflow machine Mitsubishi Sato, Yuetsu Kodama, Shuichi Sakai, Yoshinori Yamaguchi, Yasuhito Koumura
April 1992 **ACM SIGARCH Computer Architecture News , Proceedings of the 19th annual international symposium on Computer architecture ISCA '92**, Volume 20 Issue 2**Publisher:** ACM PressFull text available: pdf(967.67 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In this paper, we present a thread-based programming model for the EM-4 hybrid dataflow machine, where parallelism and synchronization among threads of sequential execution are described explicitly by the programmer. Although EM-4 was originally designed as a dataflow machine, we demonstrate that it provides effective architectural support for a variety of programming styles, including message passing and distributed data sharing in imperative languages. Our approach allows the programmer to ...

62 Time weaver: a software-through-models framework for embedded real-time systems Dionisio de Niz, Raj Rajkumar
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7**Publisher:** ACM PressFull text available: pdf(467.76 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Embedded real-time systems are deployed in a wide range of application domains including transportation systems, automated manufacturing, process control, defense, aerospace, and telecommunications. These systems must satisfy not only logical functional requirements but also *para-functional* properties such as timeliness, Quality of Service (QoS) and reliability. The cross-cutting behaviors imposed by these para-functional properties and dependencies on operational characteristics (e.g. ha ...

Keywords: couplers, embedded, real-time, semantic dimension, semantic separation, software-through-models**63 The Amber system: parallel programming on a network of multiprocessors** J. Chase, F. Amador, E. Lazowska, H. Levy, R. Littlefield
November 1989 **ACM SIGOPS Operating Systems Review , Proceedings of the twelfth ACM symposium on Operating systems principles SOSP '89**, Volume 23 Issue 5**Publisher:** ACM PressFull text available: pdf(1.53 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes a programming system called Amber that permits a single application program to use a homogeneous network of computers in a uniform way,

making the network appear to the application as an integrated multiprocessor. Amber is specifically designed for high performance in the case where each node in the network is a shared-memory multiprocessor. Amber shows that support for loosely-coupled multiprocessing can be efficiently realized using an obje ...

64 Generation and quantitative evaluation of dataflow clusters

 Lucas Roh, Walid A. Najjar, A. P. Wim Böhm

July 1993 **Proceedings of the conference on Functional programming languages and computer architecture FPCA '93**

Publisher: ACM Press

Full text available:  pdf(993.02 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

65 A Model for the Coanalysis of Hardware and Software Architectures

Fred Rose, Todd Carpenter, Sanjaya Kumar, John Shackleton, Todd Steeves Honeywell

March 1996 **Proceedings of the 4th International Workshop on Hardware/Software Co-Design CODES '96**

Publisher: IEEE Computer Society

Full text available:  pdf(1.13 MB)  Additional Information: [full citation](#), [abstract](#)
[Publisher Site](#)

Successful multiprocessor system design for complex real-time embedded applications requires powerful and comprehensive, yet cost-effective, productive, and maintain able modeling. The multi-disciplinary, VHDL-based modeling library developed by the Honeywell Technology Center places heavy emphasis on multiprocessing and distributed communications. These models focus on detailed hardware performance analysis along with multiple abstraction levels for software representation and evaluation. This ...

Keywords: VHDL, performance modeling, hardware/software codesign, RASSP

66 Untrusted hosts and confidentiality: secure program partitioning

 Steve Zdancewic, Lantian Zheng, Nathaniel Nystrom, Andrew C. Myers

October 2001 **ACM SIGOPS Operating Systems Review , Proceedings of the eighteenth ACM symposium on Operating systems principles SOSP '01**, Volume 35 Issue 5

Publisher: ACM Press

Full text available:  pdf(1.36 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents secure program partitioning, a language-based technique for protecting confidential data during computation in distributed systems containing mutually untrusted hosts. Confidentiality and integrity policies can be expressed by annotating programs with security types that constrain information flow; these programs can then be partitioned automatically to run securely on heterogeneously trusted hosts. The resulting communicating subprograms collectively implement the original p ...

67 Concepts and paradigms of object-oriented programming

 Peter Wegner

August 1990 **ACM SIGPLAN OOPS Messenger**, Volume 1 Issue 1

Publisher: ACM Press

Full text available:  pdf(5.52 MB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

We address the following questions for object-oriented programming: *What is it? What are its goals? What are its origins? What are its paradigms? What are its design alternatives? What are its models of concurrency? What are its formal computational models? What comes after object-oriented programming?* Starting from software engineering goals, we examine the origins and paradigms of object-oriented programming, explore its language design alternativ ...

68 The performance of μ-kernel-based systems

Hermann Härtig, Michael Hohmuth, Jochen Liedtke, Sebastian Schönberg, Jean Wolter

October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth**

Publisher: ACM Press

Full text available: [pdf\(2.02 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

69 Processor allocation policies for message-passing parallel computers

Cathy McCann, John Zahorjan

May 1994 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1994 ACM SIGMETRICS conference on Measurement and modeling of computer systems SIGMETRICS '94**, Volume 22 Issue 1

Publisher: ACM Press

Full text available: [pdf\(1.50 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

When multiple jobs compete for processing resources on a parallel computer, the operating system kernel's processor allocation policy determines how many and which processors to allocate to each. In this paper we investigate the issues involved in constructing a processor allocation policy for large scale, message-passing parallel computers supporting a scientific workload. We make four specific contributions: We define the concept of efficiency preservat ...

70 Research papers: test & analysis I: Automated, contract-based user testing of

commercial-off-the-shelf components

Lionel C. Briand, Yvan Labiche, Michal M. Sówka

May 2006 **Proceeding of the 28th international conference on Software engineering ICSE '06**

Publisher: ACM Press

Full text available: [pdf\(173.17 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Commercial-off-the-Shelf (COTS) components provide a means to construct software (component-based) systems in reduced time and cost. In a COTS component software market there exist component vendors (original developers of the component) and component users (developers of the component-based systems). The former provide the component to the user without source code or design documentation, and as a result it is difficult for the latter to adequately test the component when deployed in their syst ...

Keywords: COTS, UML, adequacy criteria, component

71 Addressing dynamic issues of program model checking

Flavio Lerda, Willem Visser

May 2001 **Proceedings of the 8th international SPIN workshop on Model checking of software SPIN '01**

Publisher: Springer-Verlag New York, Inc.

Full text available: [pdf\(213.42 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Model checking real programs has recently become an active research area. Programs however exhibit two characteristics that make model checking difficult: the complexity of their state and the dynamic nature of many programs. Here we address both these issues within the context of the Java PathFinder (JPF) model checker. Firstly, we will show how the state of a Java program can be encoded efficiently and how this encoding can be exploited to improve model checking. Next we show how to use sym ...

72 Architecture: Fast and fair: data-stream quality of service

Thomas Y. Yeh, Glenn Reinman

September 2005 **Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems CASES '05**

Publisher: ACM Press

Full text available: [pdf\(318.10 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Chip multiprocessors have the potential to exploit thread level parallelism, particularly in the context of embedded server farms where the available number of threads can be quite high. Unfortunately, both per-core and overall throughput are significantly impacted

by the organization of the lowest level on-chip cache. On-chip caches for CMPs must be able to handle the increased demand and contention of multiple cores. To complicate the problem, cache demand changes dynamically with phases chang ...

Keywords: CMP, NUCA, PDAS, QOS, adaptive, bandwidth, cache, chip multiprocessor, cluster, data-stream, distributed, embedded, memory wall, migration, non-uniform access, partition, per thread degradation, phase

73 Research sessions: data integration: Adapting to source properties in processing

data integration queries

Zachary G. Ives, Alon Y. Halevy, Daniel S. Weld

June 2004 **Proceedings of the 2004 ACM SIGMOD international conference on Management of data SIGMOD '04**

Publisher: ACM Press

Full text available:  pdf(197.27 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

An effective query optimizer finds a query plan that exploits the characteristics of the source data. In data integration, little is known in advance about sources' properties, which necessitates the use of *adaptive* query processing techniques to adjust query processing on-the-fly. Prior work in adaptive query processing has focused on compensating for delays and adjusting for mis-estimated cardinality or selectivity values. In this paper, we present a generalized architecture for adaptiv ...

74 Balancing register allocation across threads for a multithreaded network processor

Xiaotong Zhuang, Santosh Pande

June 2004 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2004 conference on Programming language design and implementation PLDI '04**, Volume 39

Issue 6

Publisher: ACM Press

Full text available:  pdf(429.85 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Modern network processors employ multi-threading to allow concurrency amongst multiple packet processing tasks. We studied the properties of applications running on the network processors and observed that their imbalanced register requirements across different threads at different program points could lead to poor performance. Many times application needs demand some threads to be more performance critical than others and thus by controlling the register allocation across threads one could impa ...

Keywords: multithreaded processor, network processor, register allocation

75 A machine independent interface for lightweight threads

Bodhisattwa Mukherjee, Greg Eisenhauer, Kaushik Ghosh

January 1994 **ACM SIGOPS Operating Systems Review**, Volume 28 Issue 1

Publisher: ACM Press

Full text available:  pdf(840.43 KB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Recently, lightweight thread libraries have become a common entity to support concurrent programming on shared memory multiprocessors. However, the disparity between primitives offered by operating systems creates a challenge for those who wish to create portable lightweight thread packages. What should be the interface between the machine-independent and machine-dependent parts of the thread library? We have implemented a portable lightweight thread library on top of Unix on a KSR-1 supercomput ...

76 Partitioning based operating system: a formal model

Lei Luo, Ming-Yuan Zhu

July 2003 **ACM SIGOPS Operating Systems Review**, Volume 37 Issue 3

Publisher: ACM Press

Full text available:  pdf(736.77 KB) Additional Information: [full citation](#), [abstract](#), [references](#)

Automated aircraft control has traditionally been divided into distinct functions that are implemented separately (e.g., autopilot, auto-throttle, flight management); each function

has its own fault-tolerant computer system, and dependencies among different functions are generally limited to the exchange of sensor and control data. A by-product of this federated architecture is that faults are strongly contained within the computer system of the function where they occur and cannot readily propa ...

77 Dynamically managing the communication-parallelism trade-off in future clustered processors

Rajeev Balasubramonian, Sandhya Dwarkadas, David H. Albonesi
May 2003 **ACM SIGARCH Computer Architecture News , Proceedings of the 30th annual international symposium on Computer architecture ISCA '03**, Volume 31 Issue 2

Publisher: ACM Press

Full text available: [pdf\(206.34 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Clustered microarchitectures are an attractive alternative to large monolithic superscalar designs due to their potential for higher clock rates in the face of increasingly wire-delay-constrained process technologies. As increasing transistor counts allow an increase in the number of clusters, thereby allowing more aggressive use of instruction-level parallelism (ILP), the inter-cluster communication increases as data values get spread across a wider area. As a result of the emergence of this tr ...

78 Impact of sharing-based thread placement on multithreaded architectures

R. Thekkath, S. J. Eggers
April 1994 **ACM SIGARCH Computer Architecture News , Proceedings of the 21ST annual international symposium on Computer architecture ISCA '94**, Volume 22 Issue 2

Publisher: IEEE Computer Society Press, ACM Press

Full text available: [pdf\(1.33 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Multithreaded architectures context switch between instruction streams to hide memory access latency. Although this improves processor utilization, it can increase cache interference and degrade overall performance. One technique to reduce the interconnect traffic is to co-locate threads that share data on the same processor. Multiple threads sharing in the cache should reduce compulsory and invalidation misses, thereby improving execution time. To test this hypothesis, we compared a variety of t ...

79 Physical Experimentation with Prefetching Helper Threads on Intel's Hyper-Threaded Processors

Dongkeun Kim, Steve Shih-wei Liao, Perry H. Wang, Juan del Cuvillo, Xinmin Tian, Xiang Zou, Hong Wang, Donald Yeung, Milind Girkar, John P. Shen

March 2004 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '04**

Publisher: IEEE Computer Society

Full text available: [pdf\(264.47 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Pre-execution techniques have received much attention as an effective way of prefetching cache blocks to tolerate the ever-increasing memory latency. A number of pre-execution techniques based on hardware, compiler, or both have been proposed and studied extensively by researchers. They report promising results on simulators that model a Simultaneous Multithreading (SMT) processor. In this paper, we apply the helper threading idea on a real multithreaded machine, i.e., Intel Pentium 4 processor with Hyp ...

80 Data and memory optimization techniques for embedded systems

P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P. G. Kjeldsberg
April 2001 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 6 Issue 2

Publisher: ACM Press

Full text available: [pdf\(339.91 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a survey of the state-of-the-art techniques used in performing data and memory-related optimizations in embedded systems. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three

important cost metrics: area, performance, and power dissipation of the resulting implementation. We first examine architecture-independent optimizations in the form of code transformations. We next cover a broad spectrum of optimizati ...

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, data cache, data optimization, high-level synthesis, memory architecture customization, memory power dissipation, register file, size estimation, survey

Results 61 - 80 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) **4** [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used thread partitioning based on cost model

Found 127,417 of 199,915

Sort results by

 Save results to a Binder

Display results

 Search Tips Open results in a new window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 81 - 100 of 200

Result page: previous

[1](#) [2](#) [3](#) [4](#) **5** [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale

81 Evaluating network processing efficiency with processor partitioning and asynchronous I/O

Tim Brecht, G. (John) Janakiraman, Brian Lynn, Vikram Saletore, Yoshio Turner
April 2006 **ACM SIGOPS Operating Systems Review , Proceedings of the 2006 EuroSys conference EuroSys '06**, Volume 40 Issue 4

Publisher: ACM PressFull text available: [pdf\(521.28 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Applications requiring high-speed TCP/IP processing can easily saturate a modern server. We and others have previously suggested alleviating this problem in multiprocessor environments by dedicating a subset of the processors to perform network packet processing. The remaining processors perform only application computation, thus eliminating contention between these functions for processor resources. Applications interact with packet processing engines (PPEs) using an asynchronous I/O (AIO) prog ...

Keywords: TCP/IP, asynchronous I/O, network processing**82** Compile/run-time support for threaded MPI execution on multiprogrammed shared memory machines

Hong Tang, Kai Shen, Tao Yang
May 1999 **ACM SIGPLAN Notices , Proceedings of the seventh ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '99**, Volume 34 Issue 8

Publisher: ACM PressFull text available: [pdf\(1.54 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

MPI is a message-passing standard widely used for developing high-performance parallel applications. Because of the restriction in the MPI computation model, conventional implementations on shared memory machines map each MPI node to an OS process, which suffers serious performance degradation in the presence of multiprogramming, especially when a space/time sharing policy is employed in OS job scheduling. In this paper, we study compile-time and run-time support for MPI by using threads and dem ...

83 Sharing and protection in a single-address-space operating system

Jeffrey S. Chase, Henry M. Levy, Michael J. Feeley, Edward D. Lazowska
November 1994 **ACM Transactions on Computer Systems (TOCS)**, Volume 12 Issue 4

Publisher: ACM PressFull text available: [pdf\(2.87 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article explores memory sharing and protection support in Opal, a single-address-space operating system designed for wide-address (64-bit) architectures. Opal threads execute within protection domains in a single shared virtual address space. Sharing is simplified, because addresses are context independent. There is no loss of protection,

because addressability and access are independent; the right to access a segment is determined by the protection domain in which a thread executes. T ...

Keywords: 64-bit architectures, capability-based systems, microkernel operating systems, object-oriented database systems, persistent storage, protection, single-address-space operating systems, wide-address architectures

84 Designing interconnection networks for multi-level packaging

 M. T. Raghunath, A. Ranade

December 1993 **Proceedings of the 1993 ACM/IEEE conference on Supercomputing Supercomputing '93**

Publisher: ACM Press

Full text available:  pdf(1.12 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

85 Functional Partitioning for Low Power Distributed Systems of Systems-on-a-chip

Yunsi Fei, Niraj K. Jha

January 2002 **Proceedings of the 2002 conference on Asia South Pacific design automation/VLSI Design ASP-DAC '02**

Publisher: IEEE Computer Society

Full text available:  pdf(145.31 KB)  Additional Information: [full citation](#), [abstract](#), [citations](#)
[Publisher Site](#)

In this paper, we present a functional partitioning method for low power real-time distributed embedded systems whose constituent nodes are systems-on-a-chip (SOCs). The system-level specification is assumed to be given as a set of task graphs. The goal is to partition the task graphs so that each partitioned segment is implemented as an SOC and the embedded system is realized as a distributed system of SOCs. Unlike most previous synthesis and partitioning tools, this technique merges partitioni ...

Keywords: functional partitioning, SOC synthesis, genetic algorithm

86 Provably efficient scheduling for languages with fine-grained parallelism

 Guy E. Blelloch, Phillip B. Gibbons, Yossi Matias

March 1999 **Journal of the ACM (JACM)**, Volume 46 Issue 2

Publisher: ACM Press

Full text available:  pdf(321.43 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Many high-level parallel programming languages allow for fine-grained parallelism. As in the popular work-time framework for parallel algorithm design, programs written in such languages can express the full parallelism in the program without specifying the mapping of program tasks to processors. A common concern in executing such programs is to schedule tasks to processors dynamically so as to minimize not only the execution time, but also the amount of space (memory) needed. Without caref ...

87 A Performance Evaluation of the Convex SPP-1000 Scalable Shared Memory

 Parallel Computer

Thomas Sterling, Daniel Savarese, Peter MacNeice, Kevin Olson, Clark Mobarry, Bruce Fryxell, Phillip Merkey

December 1995 **Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM) - Volume 00 Supercomputing '95**

Publisher: ACM Press

Full text available:  pdf(457.11 KB)  html(2.67 KB)  ps(780.23 KB) Additional Information: [full citation](#), [references](#), [citations](#)

88

Fine-grain multithreading with minimal compiler support—a cost effective approach to

implementing efficient multithreading languages

Kenjiro Taura, Akinori Yonezawa

May 1997 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1997 conference on Programming language design and implementation PLDI '97**, Volume 32

Issue 5

Publisher: ACM Press

Full text available:  pdf(2.03 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

It is difficult to map the execution model of multithreading languages (languages which support fine-grain dynamic thread creation) onto the single stack execution model of C. Consequently, previous work on efficient multithreading uses elaborate frame formats and allocation strategy, with compilers customized for them. This paper presents an alternative cost-effective implementation strategy for multithreading languages which can maximally exploit current sequential C compilers. We identify a s ...

89 Program analysis: Controlling factors in evaluating path-sensitive error detection techniques 

 Matthew B. Dwyer, Suzette Person, Sebastian Elbaum

November 2006 **Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering SIGSOFT '06/FSE-14**

Publisher: ACM Press

Full text available:  pdf(300.09 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Recent advances in static program analysis have made it possible to detect errors in applications that have been thoroughly tested and are in wide-spread use. The ability to find errors that have eluded traditional validation methods is due to the development and combination of sophisticated algorithmic techniques that are embedded in the implementations of analysis tools. Evaluating new analysis techniques is typically performed by running an analysis tool on a collection of subject programs, p ...

Keywords: empirical study, model checking, path-sensitive analysis

90 Adaptive two-level thread management for fast MPI execution on shared memory machines 

 Kai Shen, Hong Tang, Tao Yang

January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '99**

Publisher: ACM Press

Full text available:  pdf(152.63 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

91 Process Partitioning for Distributed Embedded Systems 

Junwei Hou, Wayne Wolf

March 1996 **Proceedings of the 4th International Workshop on Hardware/Software Co-Design CODES '96**

Publisher: IEEE Computer Society

Full text available:  pdf(784.00 KB)

Additional Information: [full citation](#), [abstract](#), [citations](#)

 Publisher Site

We present a new technique for partitioning processes in distributed embedded systems. Our heuristic algorithm minimizes both context switch and communication overhead under real-time deadline and process size constraints; it also tries to allocate functions to processors which are well-suited to that function. The algorithm analyzes the sensitivity of the latency of the task graph to changes in vertices hierarchical clustering, splitting and border adjusting. This algorithm can be used for init ...

92 JMTP: an architecture for exploiting concurrency in embedded Java applications with real-time considerations 

Rachid Helaihel, Kunle Olukotun

November 1999 **Proceedings of the 1999 IEEE/ACM international conference on Computer-aided design ICCAD '99**

Publisher: IEEE Press

Full text available:  pdf(139.94 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Using Java in embedded systems is plagued by problems of limited runtime performance and unpredictable runtime behavior. The Java Multi-Threaded Processor (JMTP) provides solutions to these problems. The JMTP architecture is a single chip containing an off-the-shelf general purpose processor core coupled with an array of Java Thread Processors (JTPs). Performance can be improved using this architecture by exploiting coarse-grained parallelism in the application. These performance im ...

93 Hierarchical model-based autonomic control of software systems

 Marin Litoiu, Murray Woodside, Tao Zheng

May 2005 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2005 workshop on Design and evolution of autonomic application software DEAS '05**, Volume 30 Issue 4

Publisher: ACM Press

Full text available:  pdf(393.76 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Various control algorithms are used in autonomic control to maintain Quality of Service (QoS) and Service Level Agreements (SLAs). Controllers are all based to some extent on models of the relationship between resources, QoS measures, and the workload imposed by the environment. This work discusses the range of algorithms with an emphasis on richer and more powerful models to describe non-linear performance relationships, and strong interactions among the system resources. A hierarchical framewo ...

Keywords: autonomic computing, performance models, self-management

94 1 - Special Section: Efficient software implementation of embedded communication protocol controllers using asynchronous software thread integration with time- and space-efficient procedure calls

 Nagendra J. Kumar, Vasanth Asokan, Siddhartha Shivshankar, Alexander G. Dean
February 2007 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 6 Issue 1

Publisher: ACM Press

Full text available:  pdf(596.75 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The overhead of context switching limits efficient scheduling of multiple concurrent threads on a uniprocessor when real-time requirements exist. A software-implemented protocol controller may be crippled by this problem. The available idle time may be too short to recover through context switching, so only the primary thread can execute during message activity, slowing the secondary threads and potentially missing deadlines. Asynchronous software thread integration (ASTI) uses coroutine calls a ...

Keywords: Asynchronous software thread integration, J1850, fine-grain concurrency, hardware to software migration, software-implemented communication protocol controllers

95 NanoFabrics: spatial computing using molecular electronics

 Seth Copen Goldstein, Mihai Budiu
May 2001 **ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on Computer architecture ISCA '01**, Volume 29 Issue 2

Publisher: ACM Press

Full text available:  pdf(996.26 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The continuation of the remarkable exponential increases in processing power over the recent past faces imminent challenges due in part to the physics of deep-submicron CMOS devices and the costs of both chip masks and future fabrication plants. A promising solution to these problems is offered by an alternative to CMOS-based computing, chemically assembled electronic nanotechnology (CAEN).

In this paper we outline how CAEN-based computing can become a reality. We briefly describe rec ...

96 Verification: Scaling model checking of dataraces using dynamic information

Ohad Shacham, Mooly Sagiv, Assaf Schuster

June 2005 **Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '05**

Publisher: ACM Press

Full text available: [pdf\(250.26 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Dataraces in multithreaded programs often indicate severe bugs and can cause unexpected behaviors when different thread interleavings are executed. Because dataraces are a cause for concern, many works have dealt with the problem of detecting them. Works based on dynamic techniques either report errors only for dataraces that occur in the current interleaving, which limits their usefulness, or produce many spurious dataraces. Works based on model checking search exhaustively for dataraces and th ...

Keywords: data race detection, datarace, lockset, model checking, multithreading

97 Balanced Multithreading: Increasing Throughput via a Low Cost Multithreading

Hierarchy

Eric Tune, Rakesh Kumar, Dean M. Tullsen, Brad Calder

December 2004 **Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture MICRO 37**

Publisher: IEEE Computer Society

Full text available: [pdf\(197.41 KB\)](#) Additional Information: [full citation](#), [abstract](#)

A simultaneous multithreading (SMT) processor can issue instructions from several threads every cycle, allowing it to effectively hide various instruction latencies; this effect increases with the number of simultaneous contexts supported. However, each added context on an SMT processor incurs a cost in complexity, which may lead to an increase in pipeline length or a decrease in the maximum clock rate. This paper presents new designs for multithreaded processors which combine a conservative SMT ...

98 Requirements interaction management

William N. Robinson, Suzanne D. Pawlowski, Vecheslav Volkov

June 2003 **ACM Computing Surveys (CSUR)**, Volume 35 Issue 2

Publisher: ACM Press

Full text available: [pdf\(1.24 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Requirements interaction management (RIM) is the set of activities directed toward the discovery, management, and disposition of critical relationships among sets of requirements, which has become a critical area of requirements engineering. This survey looks at the evolution of supporting concepts and their related literature, presents an issues-based framework for reviewing processes and products, and applies the framework in a review of RIM state-of-the-art. Finally, it presents seven research ...

Keywords: KAOS, KATE, Oz, Requirements engineering, Telos, WinWin, analysis and design, composite system, deficiency driven design, dependency analysis, distributed intentionality, interaction analysis, software cost reduction (SCR), system architecture, system specification, viewpoints

99 Integrated document caching and prefetching in storage hierarchies based on

Markov-chain predictions

Achim Kraiss, Gerhard Weikum

August 1998 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 7 Issue 3

Publisher: Springer-Verlag New York, Inc.

Full text available: [pdf\(603.01 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Large multimedia document archives may hold a major fraction of their data in tertiary storage libraries for cost reasons. This paper develops an integrated approach to the vertical data migration between the tertiary, secondary, and primary storage in that it reconciles speculative prefetching, to mask the high latency of the tertiary storage, with the replacement policy of the document caches at the secondary and primary storage level, and also considers the interaction of these policies with ...

Keywords: Caching, Markov chains, Performance, Prefetching, Scheduling, Stochastic modeling, Tertiary storage

100 BOS is boss: a case for bulk-synchronous object systems

 Mark W. Goudreau, Kevin Lang, Girija Narlikar, Satish B. Rao

June 1999 **Proceedings of the eleventh annual ACM symposium on Parallel algorithms and architectures SPAA '99**

Publisher: ACM Press

Full text available:  [pdf\(1.42 MB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)

Results 81 - 100 of 200

Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) **5** [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)